CLARAty Reusable Robotic Software

Coupled Layer Architecture for Robotic Autonomy

Issa A.D. Nesnas
Group Supervisor, Robotic Software Systems
Mobility and Robotics Section
Jet Propulsion Laboratory

and the CLARAty team

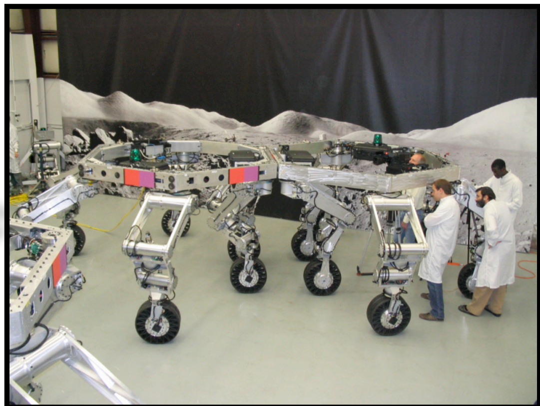
Section Seminar
July 25, 2007

Presentation Overview

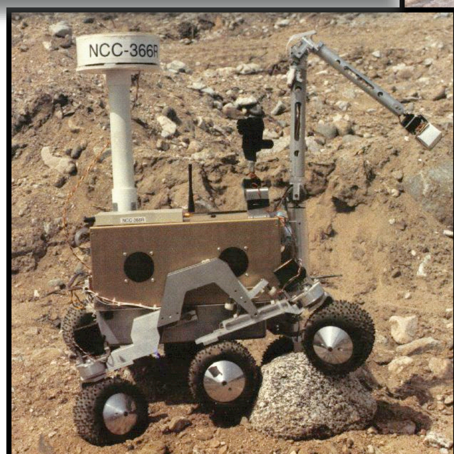
- **Problem Statement**
- **Sample Deployments** (videos)
- **Challenges**
- **Technical Approach**
- **Status**
- **Summary**

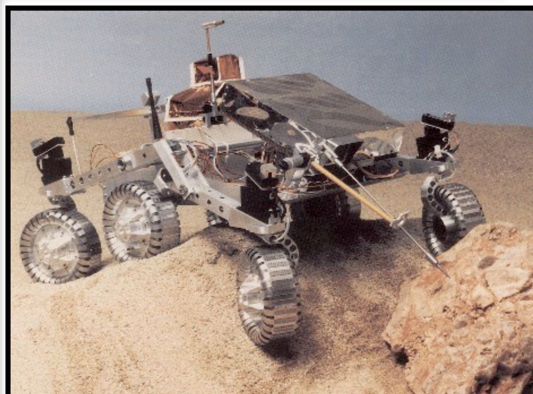
NASA/JPL Developed Different Rovers

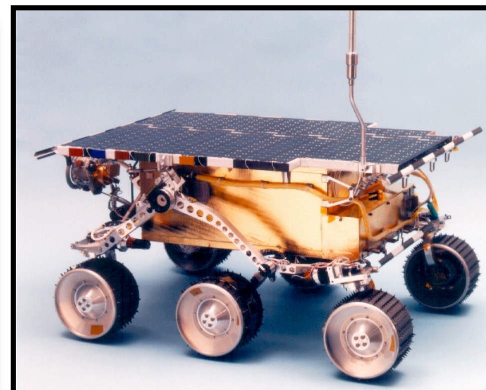
Large



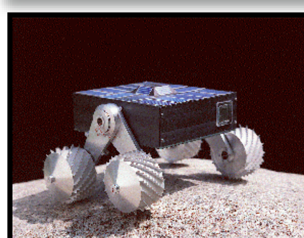
Medium







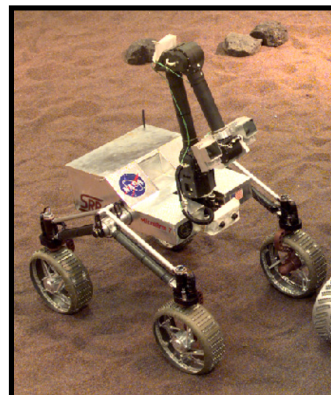
Small

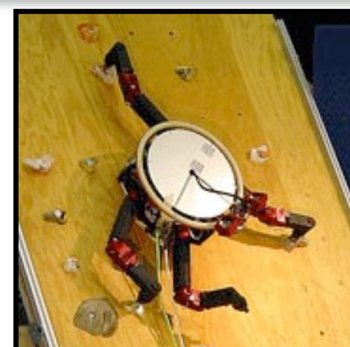


Research & flight

Different Mobility Mechanisms

- Wheeled vehicles
- Legged vehicles
- Hybrid vehicles
- Different wheel types and configuration





Problem and Challenges

- **Problem:**
  - Difficult to **share algorithms across projects**
  - **Lack** of **integrated** and **validated** robotic technologies prior to flight infusion
  - **Redundant** infrastructure for robotic projects / platforms
  - **No framework** to capture technologies from universities and other centers
  - **No interoperable** software among robotic platforms (e.g. Rocky 8, FIDO, Rocky 7, K9, K10, ATRV)
- **Key Challenges**
  - Robots have different physical characteristics
  - Robots have different hardware architectures
  - Contributions made by multiple institutions
  - Advanced research requires a flexible framework
  - Software must support various platforms
  - Lack of a common low-cost robotic platforms
  - Software must be unrestricted and accessible (ITAR and IP)
  - Software must integrate legacy code bases

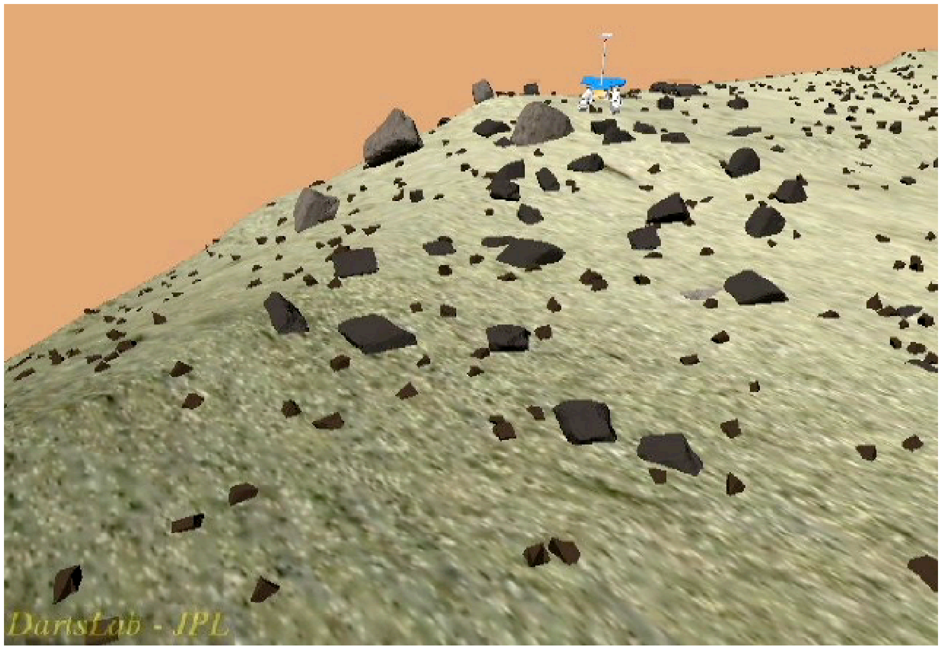
CLARAty Reusable Robotic Software

- Initiated by the Mars Technology Program (MTP) in 1999 to unify its robotic software developments
- Collaborative effort among **JPL**, **NASA ARC** and **Carnegie Mellon**
- Supported two rounds of MTP competed research programs
- Supported formal technology validation for MSL Focused Technology Program

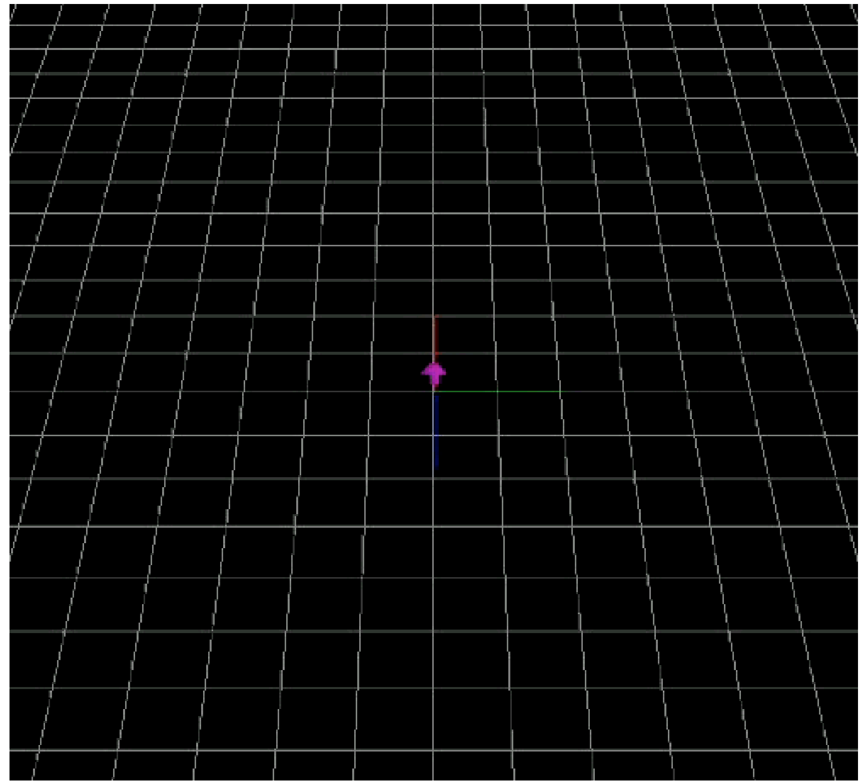
Sample Deployments

- Navigating in a Simulated Environment
- Navigating Multiple Rovers in Mars Yard
- Autonomous Targeted Driving
- Single-cycle Instrument Placement
- Targeted Driving with MER and MSL algorithms
- Advanced Locomotion and Path Planning

Navigating in a Simulated Environment



ROAMS



CLARAty Morphin
Navigator GUI

Courtesy of SOOPS task

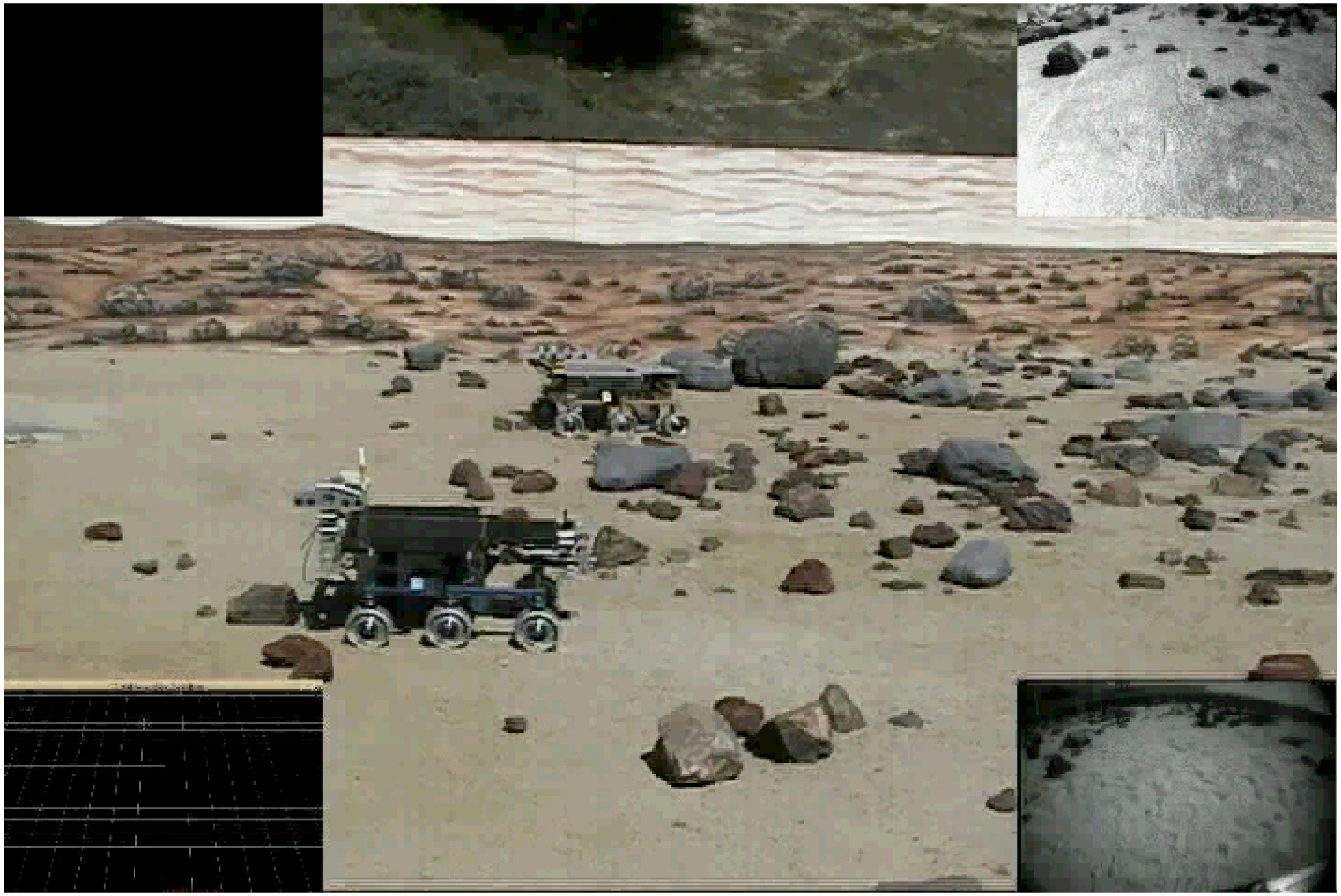
Sample Deployments

- Navigating in a Simulated Environment
- Navigating Multiple Rovers in Mars Yard
- Autonomous Targeted Driving
- Single-cycle Instrument Placement
- Targeted Driving with MER and MSL algorithms
- Advanced Locomotion and Path Planning

Navigation with Path Planning on Two Rovers

Complex Algorithms on different Platforms

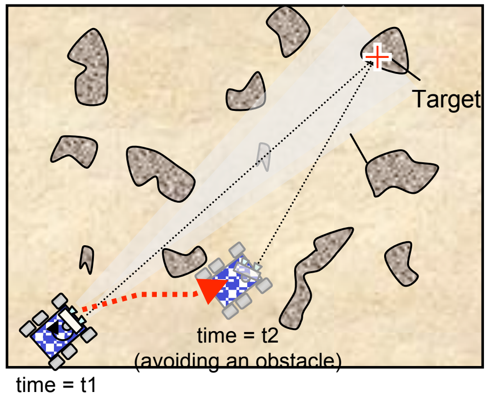
- I/O, motion control
- Trajectory Generation
- Rough Terrain Locomotion
- Odometry Pose Estimation
- Stereo Processing
- Visual Odometry
- Navigation (Morphin)
  - Obstacle avoidance
  - Path Planning



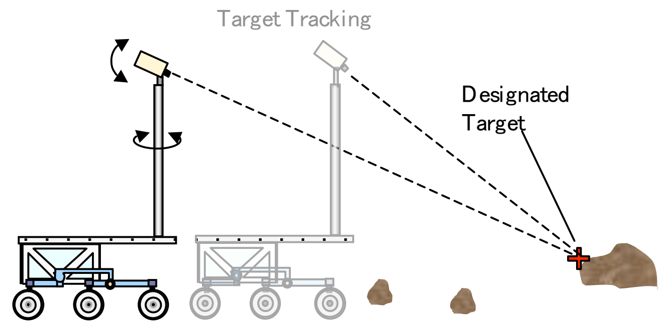
Sample Deployments

- Navigating in a Simulated Environment
- Navigating Multiple Rovers in Mars Yard
- Autonomous Targeted Driving
- Single-cycle Instrument Placement
- Targeted Driving with MER and MSL algorithms
- Advanced Locomotion and Path Planning

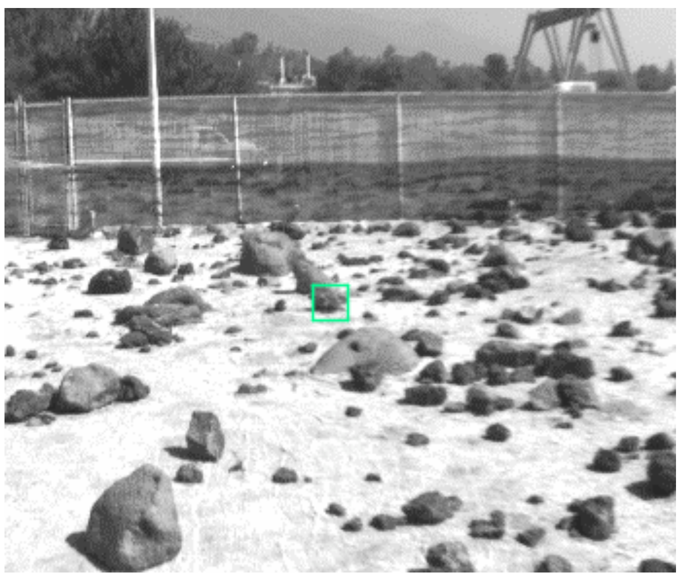
Targeted Driving using Visual Tracking



(a)

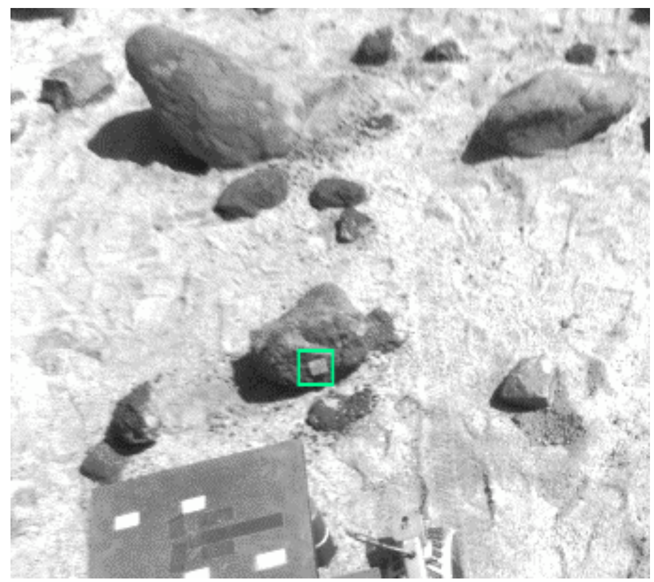


(b)



1st Frame

Changes in FOV



37th Frame after 10 m

Autonomous Targeted Driving
(more difficult terrain)

Uses following technologies:

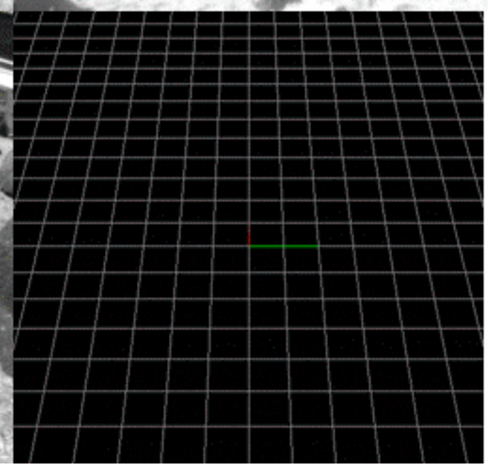
- Visual Target Tracking
  - **Falcon** or
  - MER VTT
- Navigation with obstacle avoidance
  - **Morphin** or
  - MER GESTALT
- Pose estimation
  - **6DOF EKF** and
  - **Visual odometry**

Autonomous Targeted Driving
(more difficult terrain)

VTT Image - navcam

VO Image - navcam

Front Hazcam



Morphin Navigator

Sample Deployments

- Navigating in a Simulated Environment
- Navigating Multiple Rovers in Mars Yard
- Autonomous Targeted Driving
- Single-cycle Instrument Placement
- Targeted Driving with MER and MSL algorithms
- Advanced Locomotion and Path Planning

Integrated Single-Cycle Instrument Placement

Sample Deployments

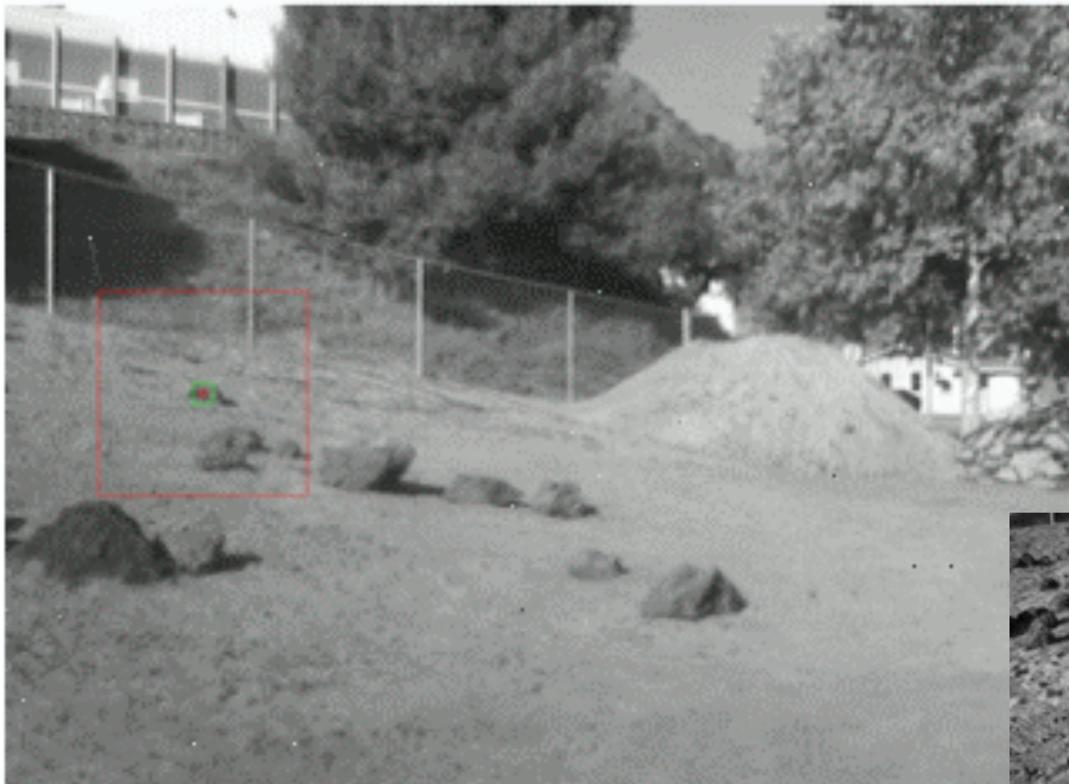
- Navigating in a Simulated Environment
- Navigating Multiple Rovers in Mars Yard
- Autonomous Targeted Driving
- Single-cycle Instrument Placement
- Targeted Driving with MER and MSL algorithms
- Advanced Locomotion and Path Planning

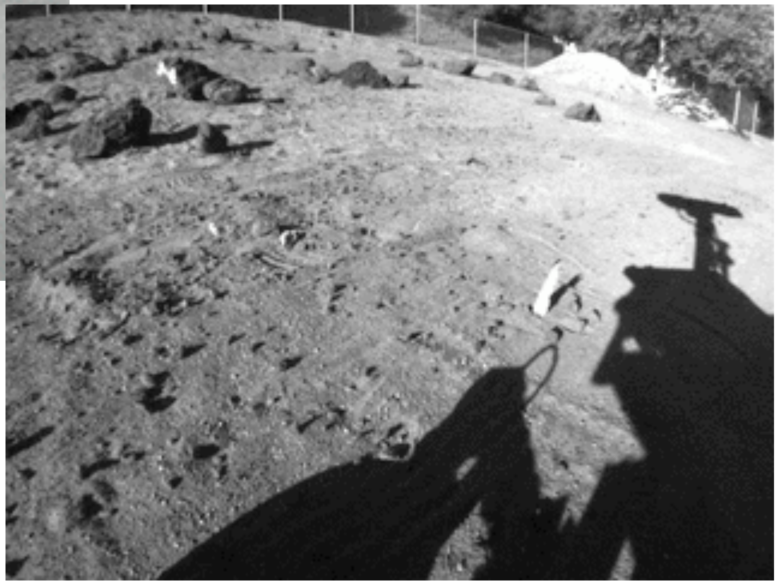
Autonomous Targeted Driving

Uses following technologies:

- Visual Target Tracking
  - Falcon or
  - **MER VTT**
- Navigation with obstacle avoidance
  - Morphin or
  - **MER GESTALT**
- Pose estimation
  - 6DOF EKF or
  - **Visual odometry**

Autonomous Targeted Driving

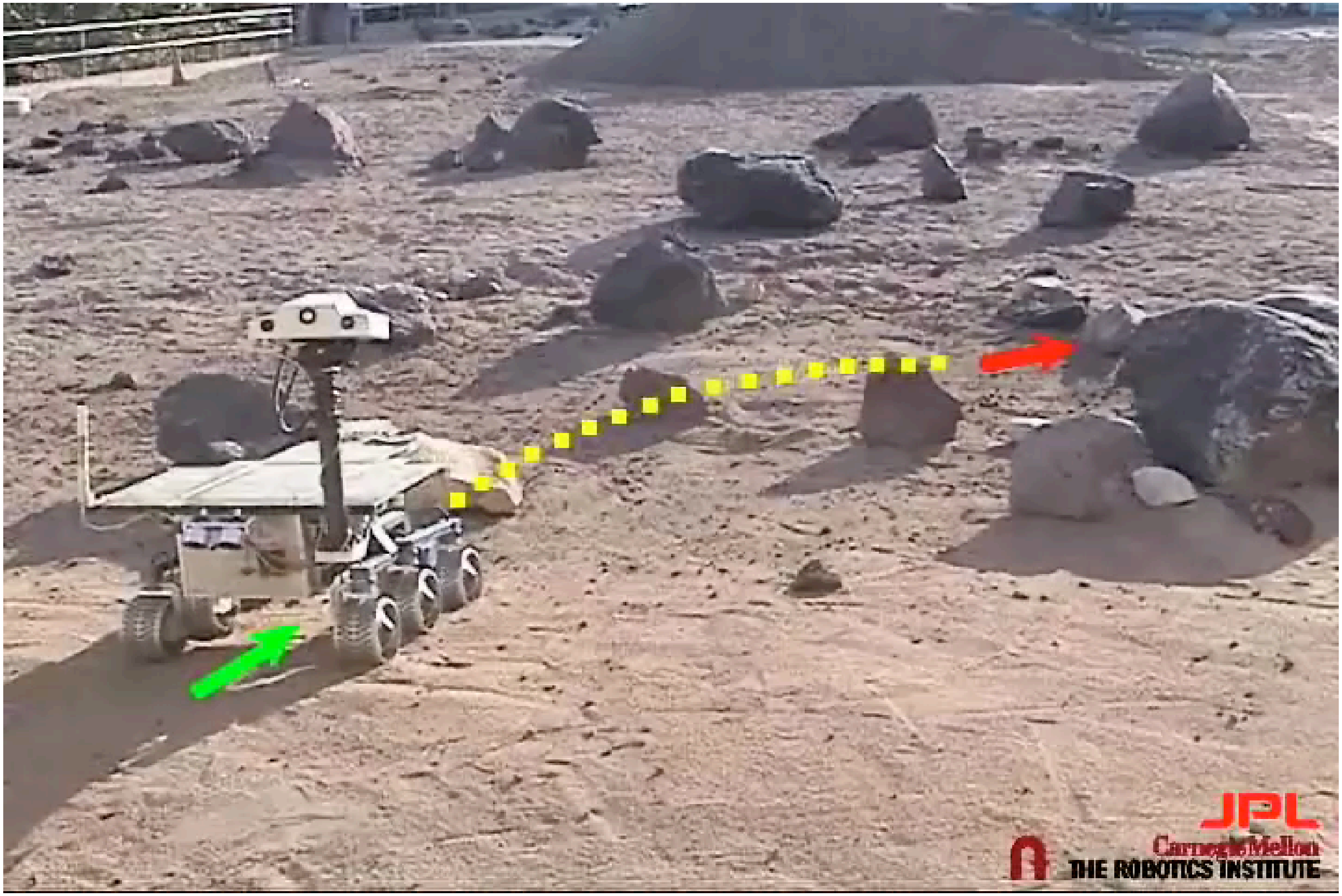




Sample Deployments

- Navigating in a Simulated Environment
- Navigating Multiple Rovers in Mars Yard
- Autonomous Targeted Driving
- Single-cycle Instrument Placement
- Targeted Driving with MER and MSL algorithms
- Advanced Locomotion and Path Planning

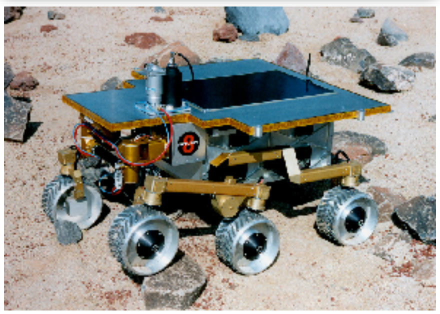
Advanced Path Planning and Locomotion

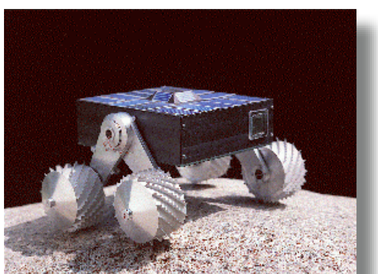


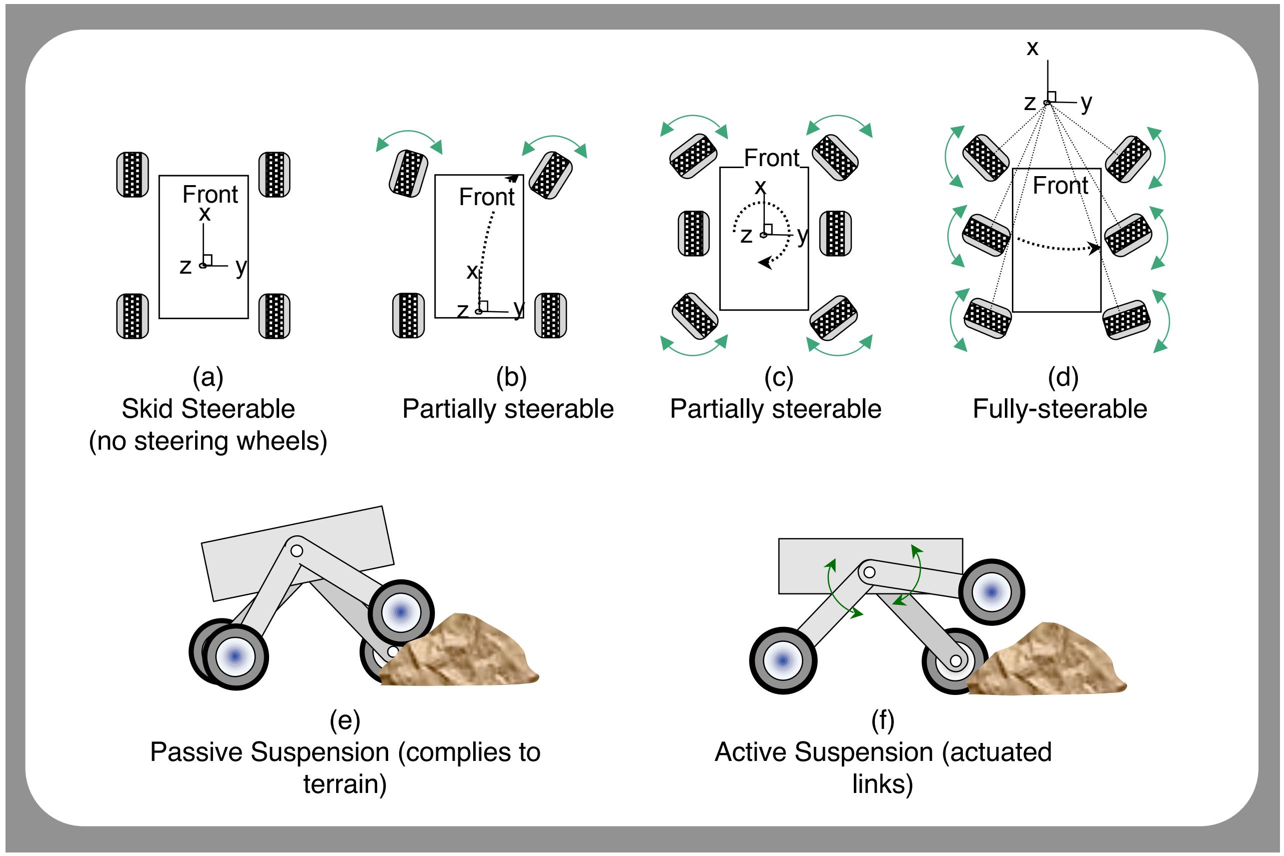
Challenges of Interoperability

- Mechanisms and Sensors
- Hardware Architecture
- Software Algorithms

Different Mobility Mechanisms

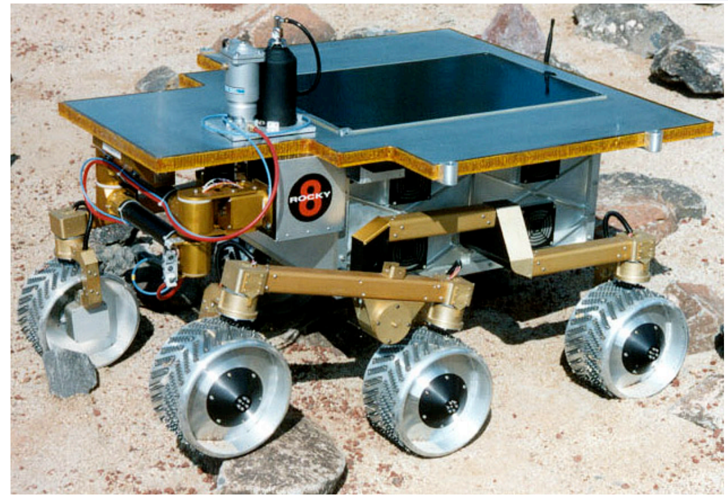




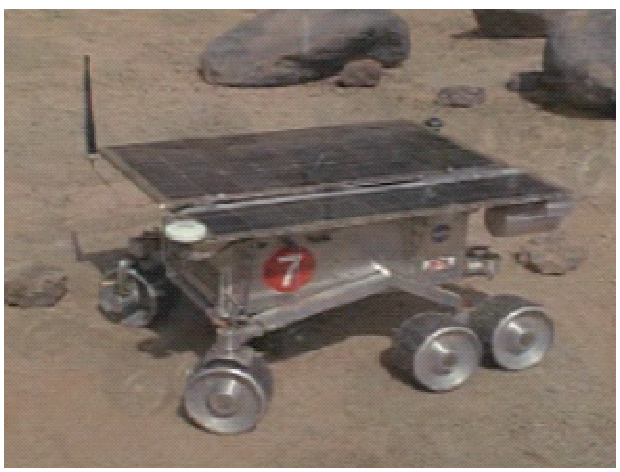


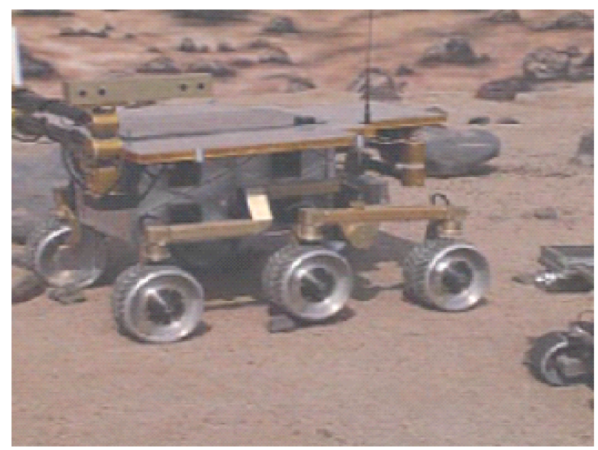
Have Different Capabilities

Rocky 7

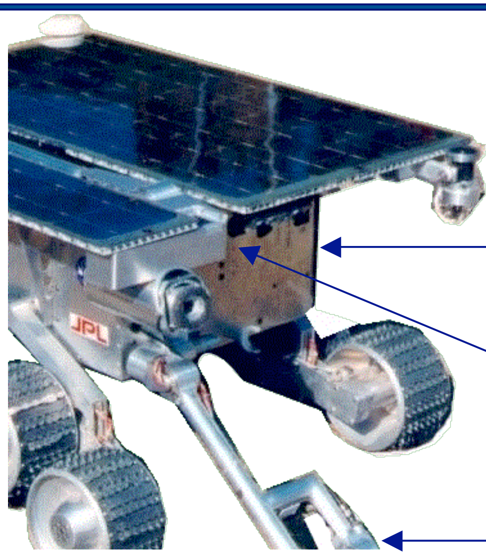


Rocky 8





Different Sensors and Appendages

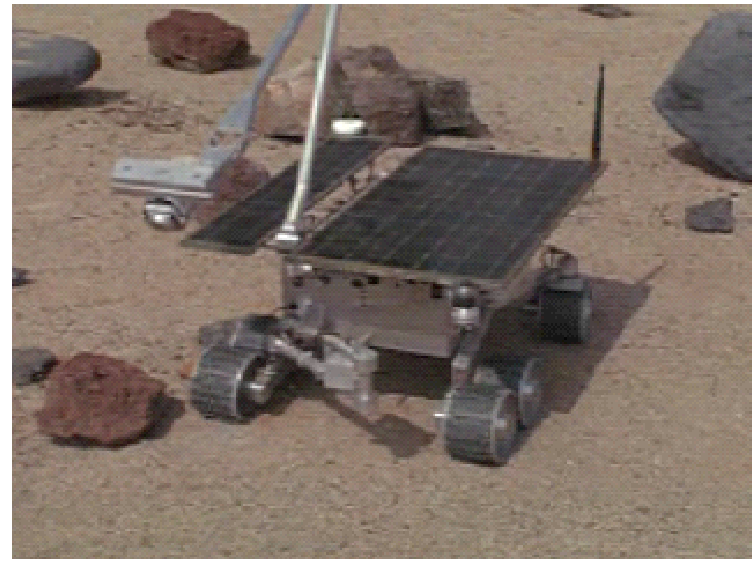


Custom Analog Sun Sensor

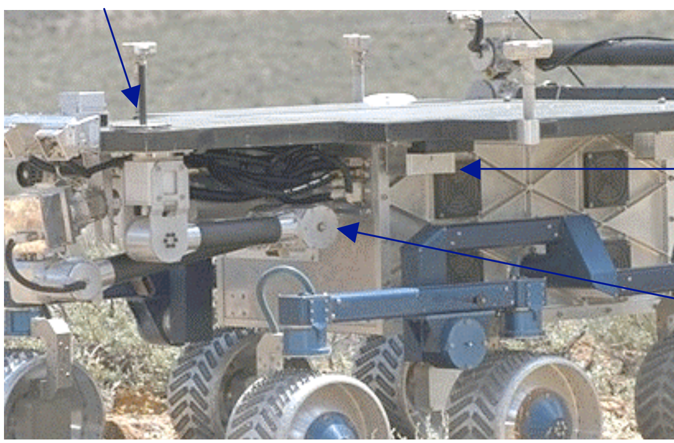
3 Accels
z-axis gryo

3 DOF Mast

2 DOF Arm



Camera Sun Sensor



4 DOF Mast

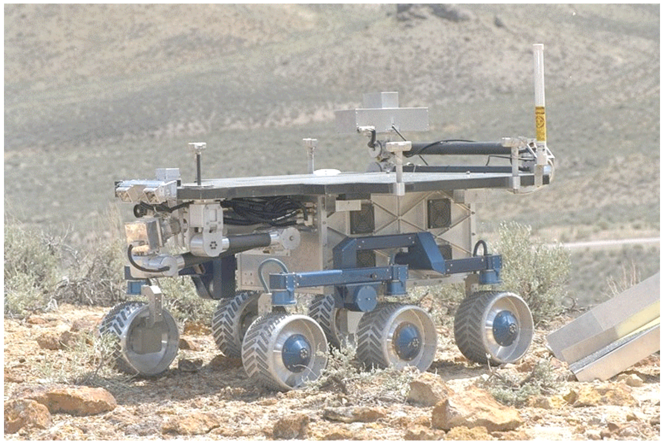
6 DOF IMU

4 DOF Arm

Challenges of Interoperability

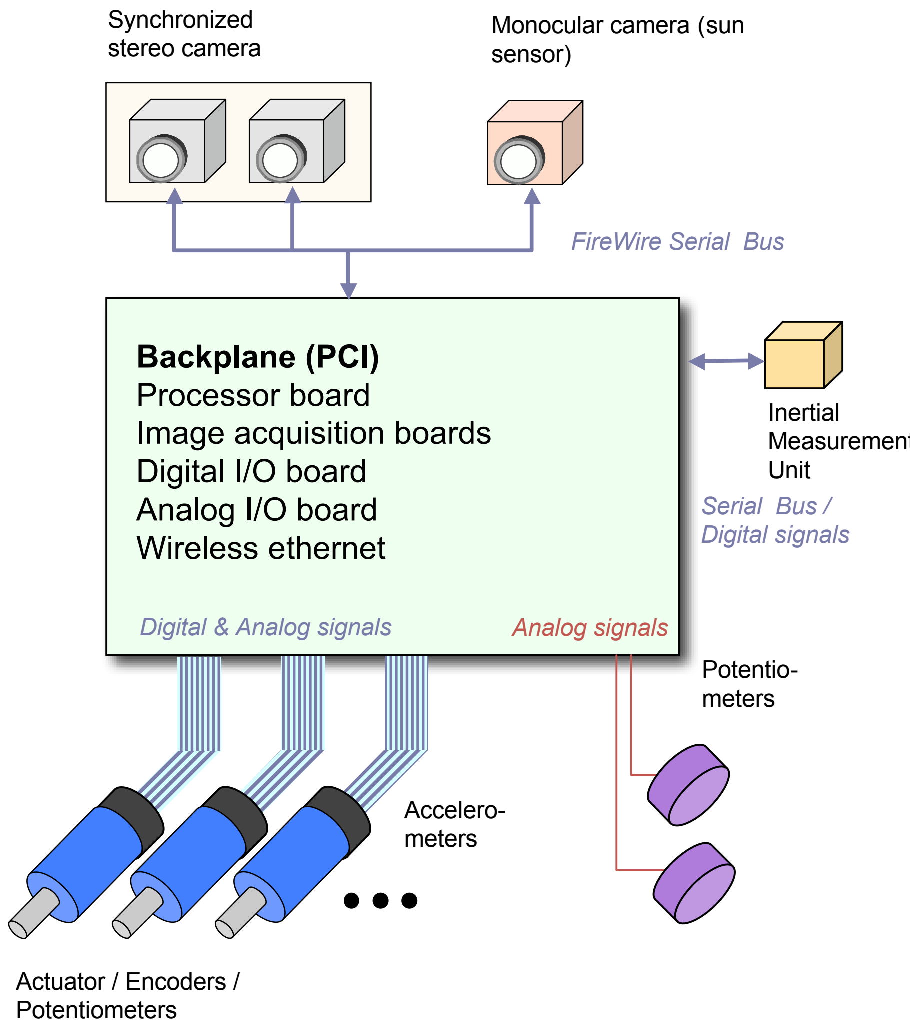
- Mechanisms and Sensors
- Hardware Architecture
- Software Algorithms

Centralized Hardware Architecture

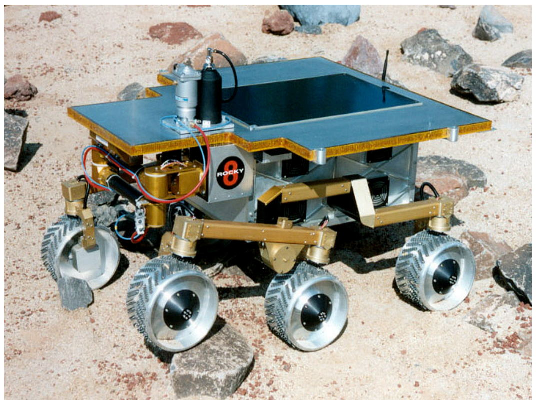


FIDO

Athena



Distributed Hardware Architecture

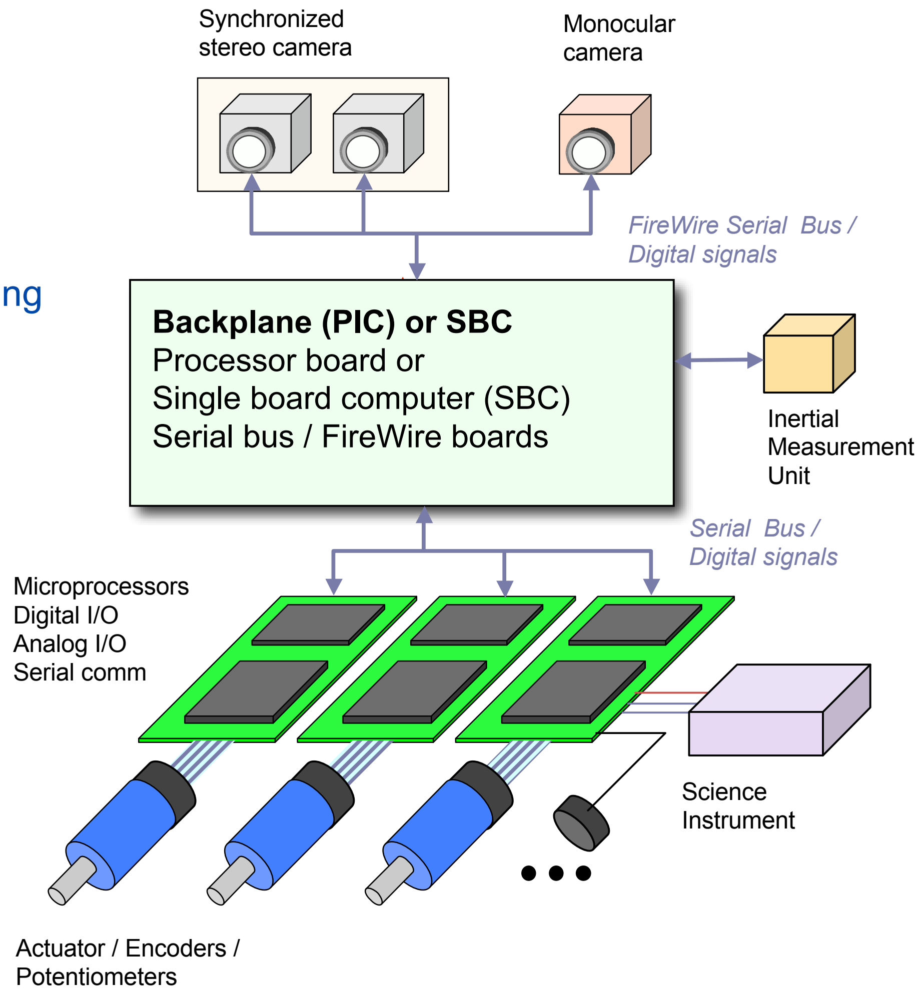


Rocky Widgets
- I2C Bus
- Single Axis Controllers
- Current Sensing
- Digital I/O
- Analog I/O

Rocky 8

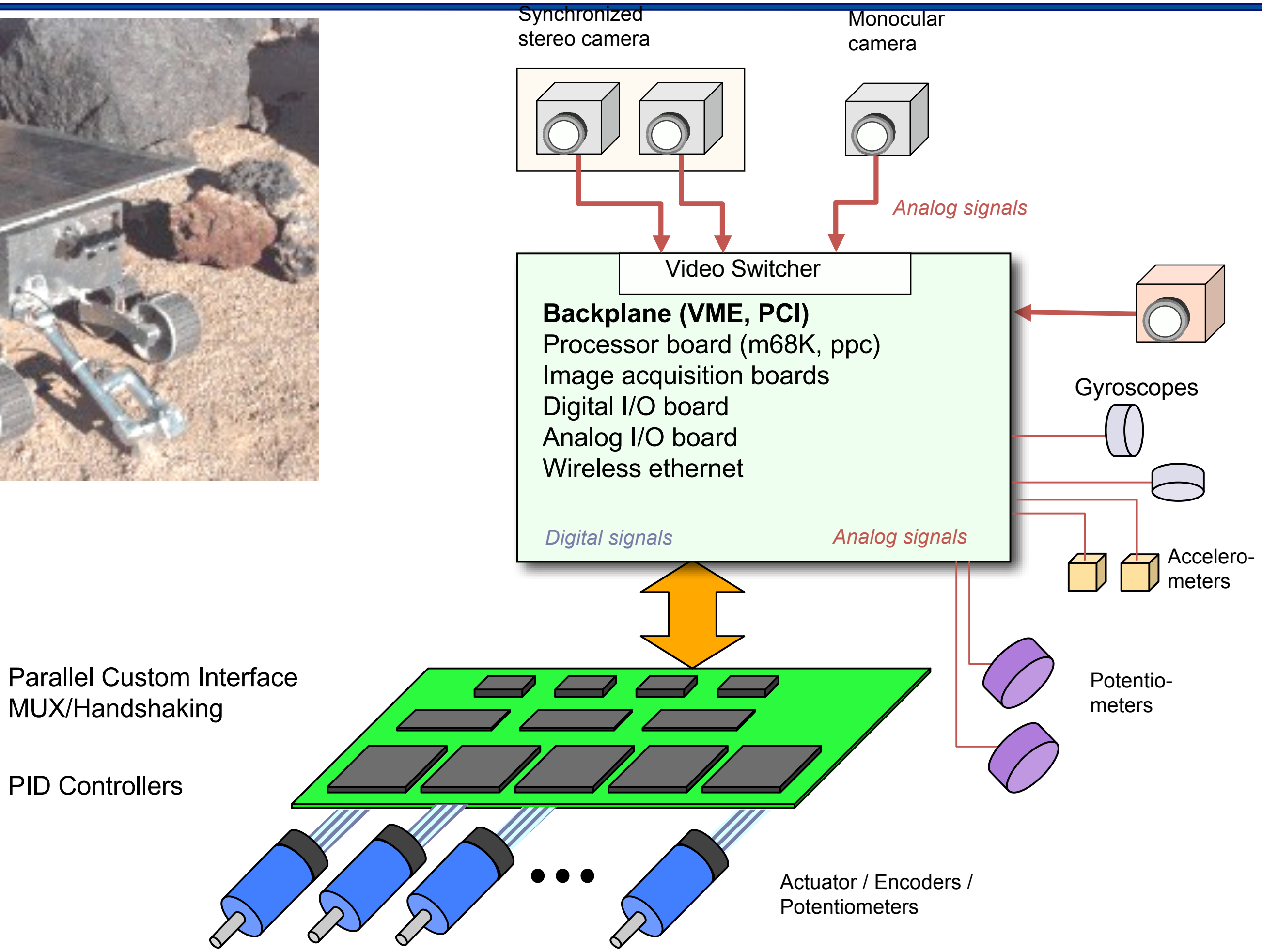
K9

K9 Controllers
- Serial Bus RS485
- PIC Servos
- Current Sensing
- Digital & Analog I/O



Hybrid Architecture

Rocky 7



Challenges of Interoperability

- Mechanisms and Sensors
- Hardware Architecture
- Software Algorithms

Software Challenges for Algorithm Infusion

The *new* algorithms to be integrated may:

- Have architectural mismatches with the framework
- Include multiple orthogonal functionalities
- Make implicit assumptions about the platform
- Duplicate functionality in the framework
- Use incompatible data structures
- Be complex and hard to tune
- Depend on specific platform
- Require highly specialized domain expertise
- Be poorly implemented

Technical Approach

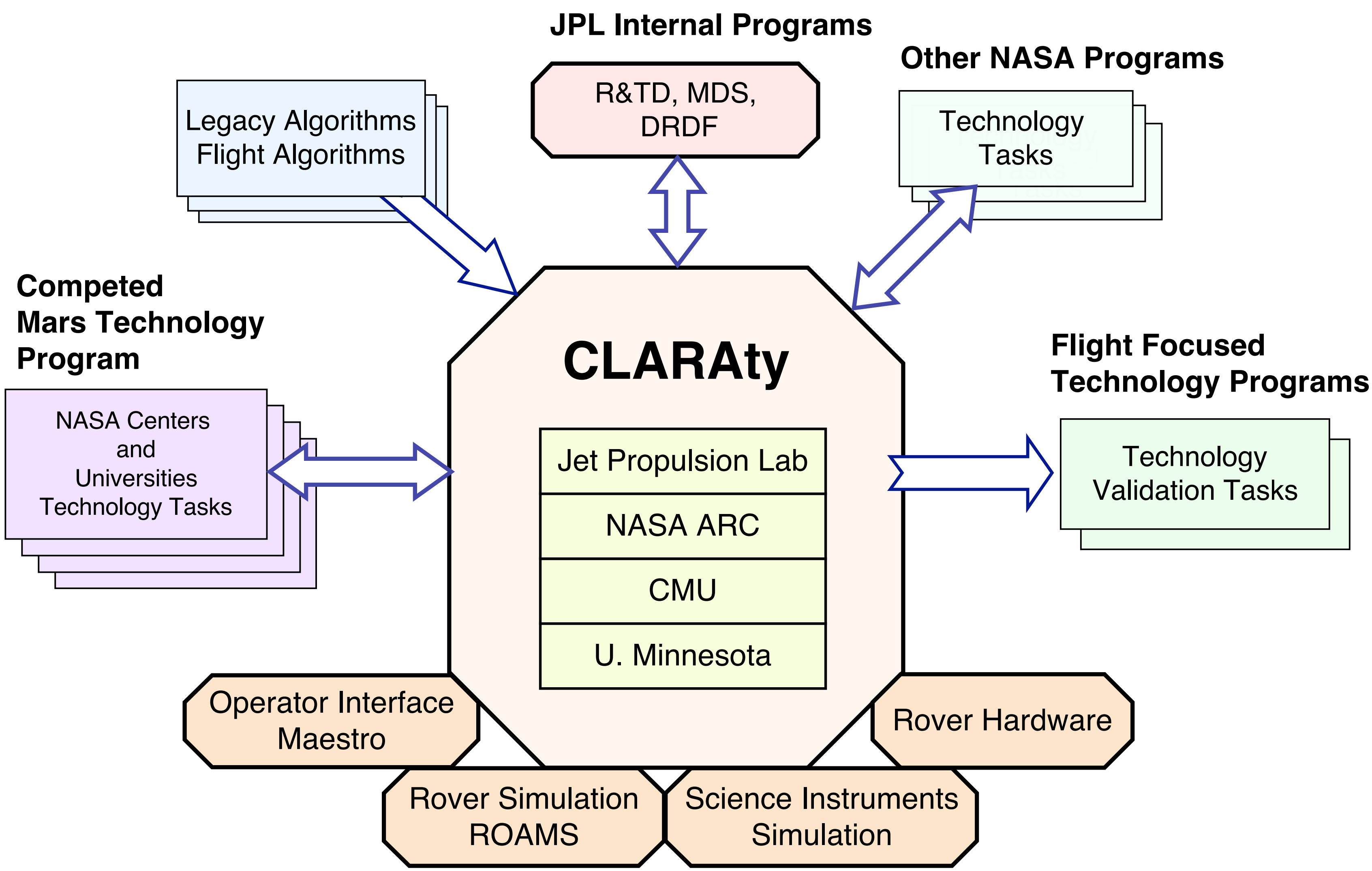
Historical Antecedents

- **Late 80's - Early 90's:** parallel robotic developments
  - RSI, MOTES, Satellite Servicing, Robby, Mircorover
  - Very limited sharing of hardware or software
- **Mid 90's:** Mars rover research centralized with Rocky 7
  - First flight rover
- **Late 90's:** Expansion and diversification of rover work
  - Very limited software interoperability (Rocky 7, FIDO, Athena, DARPA)
  - Autonomy demonstration of Remote Agent Experiment (ARC and JPL)
  - MDS investigates reusable software for spacecraft control.
- **'99-Early 00:** Mars Technology Program envisions a unified autonomy architecture
  - Funds effort to unify autonomy and robotic control
  - Starts the CLARAty task in December 1999
- **Early 00's:** Development and deployment of common robotic software
  - Started development of a shared robotic software infrastructure among JPL, NASA ARC and Carnegie Mellon
  - CLARAty supports MTP's competed research program (01-03)
  - CLARAty supports Mars Science Laboratory Focused Technology validation program
- **Mid 00's:**
  - CLARAty supports MTP's NRA program (03-06) and other NASA programs
  - CLARAty completes its first public release

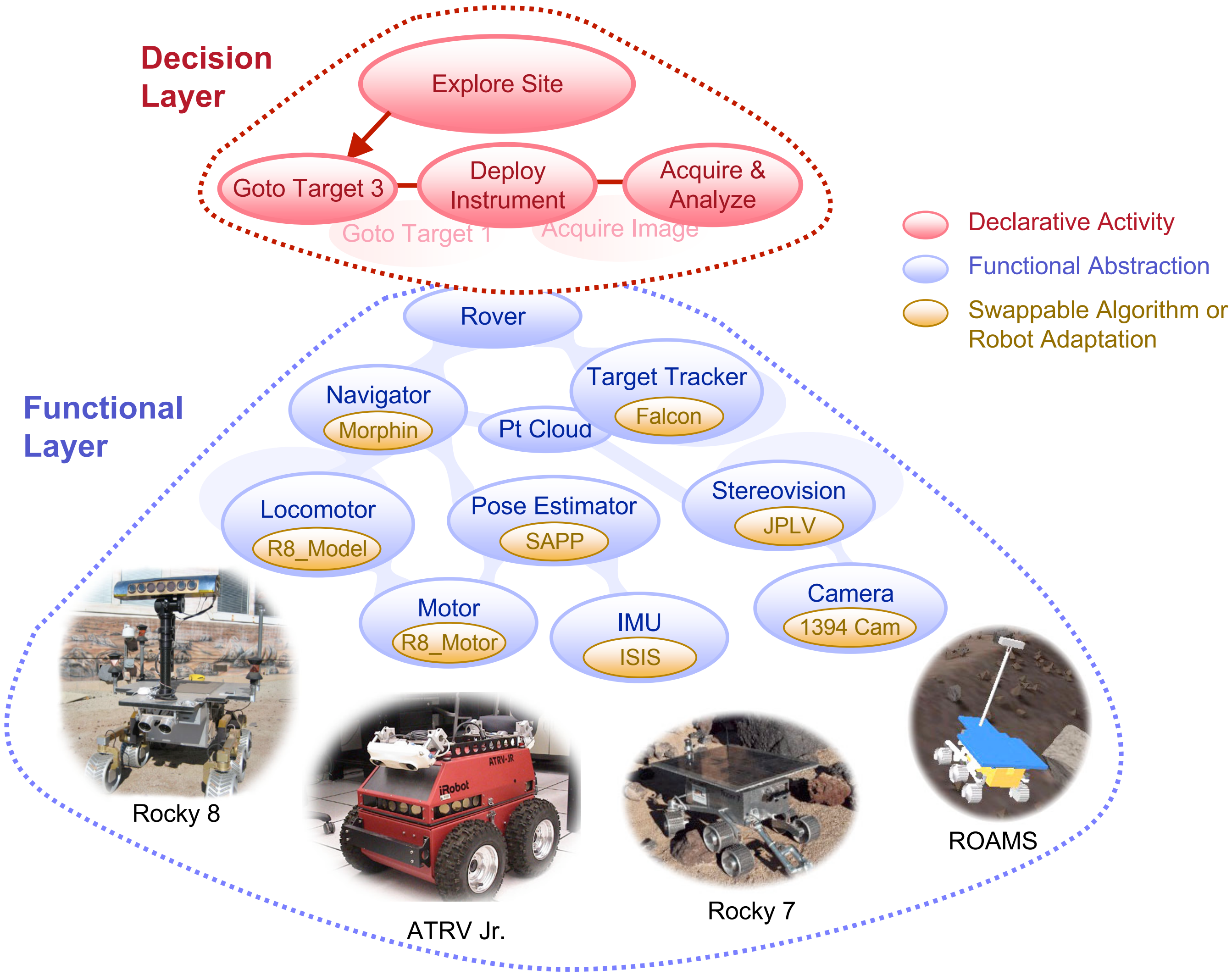
State-of-the-Art in the Robotics Community

- **Within NASA:**
  - NASREM - temporal based decomposition. Evolved into 4D-RCS currently developed at NIST
  - ESMD funded a multi-center effort (JTARS) which followed the DoD JAUS model
- **Outside NASA**
  - JAUS
    - DoD/DARPA integration environment; Scope - all unmanned military vehicle; high level message-set based
  - RosTa
    - Funded by the EU and led by Fraunhofer in Germany; started in 2007
  - Player/Stage
    - Developed at USC; client server architecture; supports multiple COTS rovers; most recognized
  - MS Robotics Studio
    - Released in 2007; service oriented architecture; XML based message passing; supports heterogeneous programming languages
  - OROCOS
    - Funded by in part by the EU and led by K.U. Leuven; started in September 2001; provides CORBA based real-time tool kit, bayesian filtering library and kinematics and dynamics
    - Similar to CLARAty's libraries and abstractions
  - LAAS
    - Led by LAAS center in France; provides a similar decomposition of Functional Level / Decision Level
    - Support DL work; FL uses a framework similar to ControlShell's component architecture
  - Miro
    - Developed at a university in Germany for the Robocup competition; corba-based real-time framework
  - Others
    - Constellation from RTI (former ControlShell), NDDS from RTI, ESRP from Evolution, ROCI from U. Penn (ROCI), OSCAR from U. Texas, MARIE from U. Sherbrooke, ARIA from MobileRobots, Mobility from Irobot
- **Comparison:**
    - Several hierarchical decomposed system; disagreement on how to decompose the hierarchy
    - Limited functionality and flexibility to support development of advanced algorithms
    - Most are constrained for supporting custom robots

Process and Collaborations



Overall Architecture



Technical Approach

- Capture **requirements** from domain experts
- Use **global perspective** across domains (motion, vision, estimation, navigation)
- Identify **recurring patterns** and **common infrastructure** therein
- Use **domain expert** to guide design
- Define **proper interfaces** for each subsystem
- Develop **generic framework** to support various implementations
- Adapt **legacy implementations** to validate framework
- **Encapsulate** when re-factoring is not feasible or affordable
- Develop **regression tests** where feasible
- **Test** on multiple robotic platforms and **study limitations**
- **Feed** learned experience **back** into the design
- **Review** and **update** to address limitations

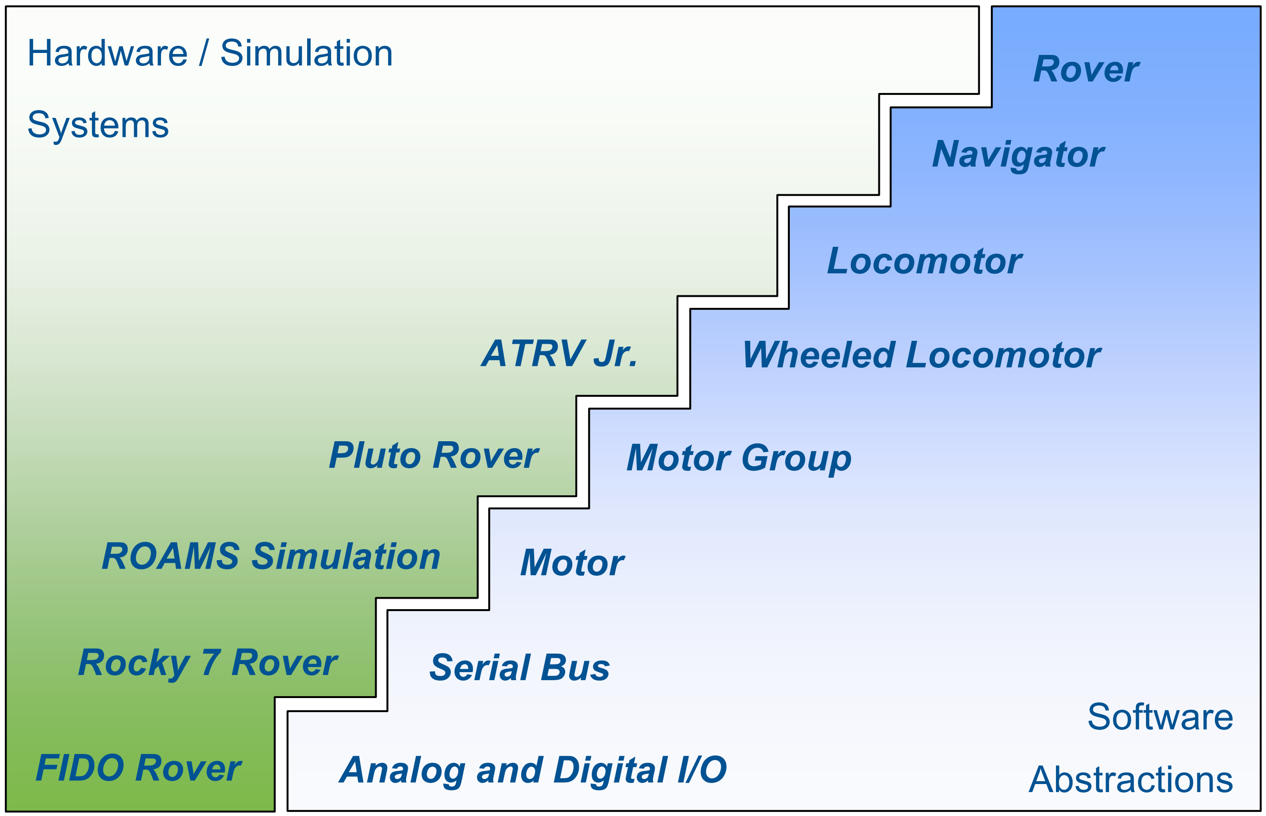
After several iterations one hopes to have achieved a truly reusable infrastructure

Declarative vs. Procedural Programming

Declarative Programming	Procedural Programming

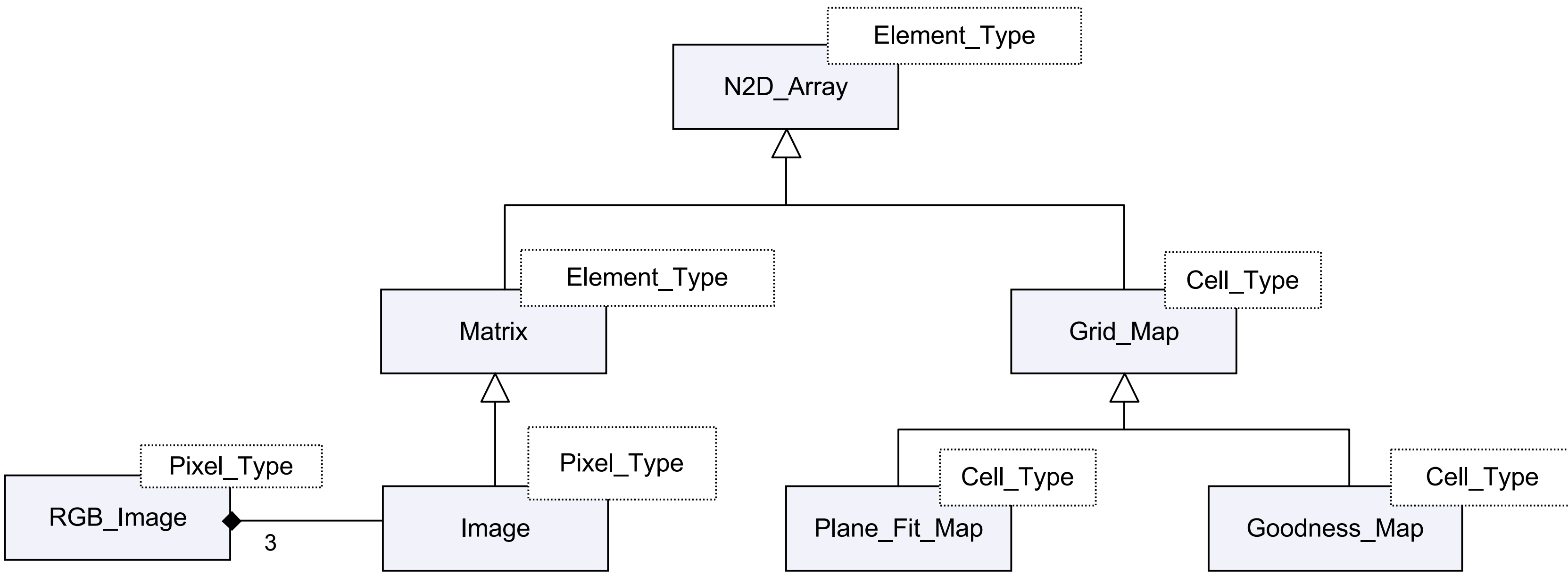
`Rover.navigate_from_to(Loc1, Loc2)`
 	
Preconditions: `near(Loc1,Loc2)`
`rover.has_power(Loc1,Loc2)`
`rover.has_time(Loc1,Loc2)`
 	
Effects: `rover.is_at(Loc2)`
	If `near(Loc1,Loc2) AND`
	`rover.has_power(Loc1,Loc2) AND`
	`rover.has_time(Loc1,Loc2) AND`
	Then: `rover.navigate_from_to(Loc1,Loc2)`

Guideline I: Multi-level Access

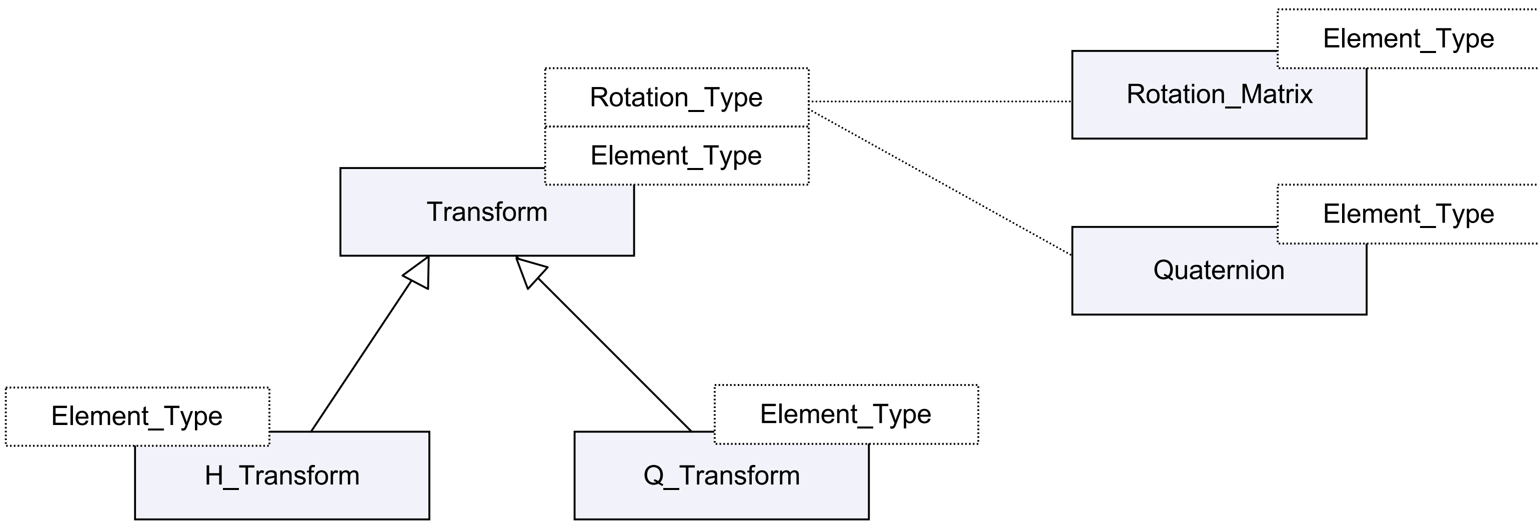


Multi-level mobility abstractions

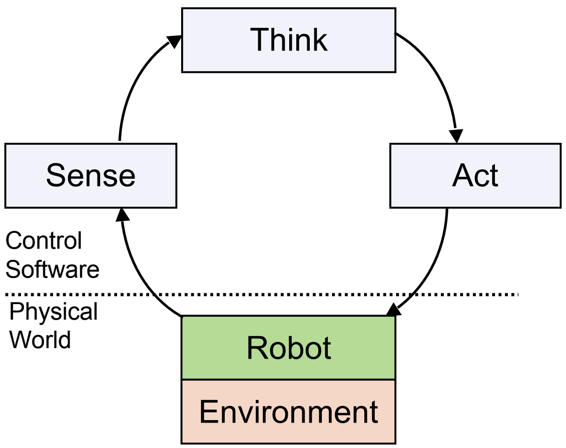
Guideline II: Common Data Structures

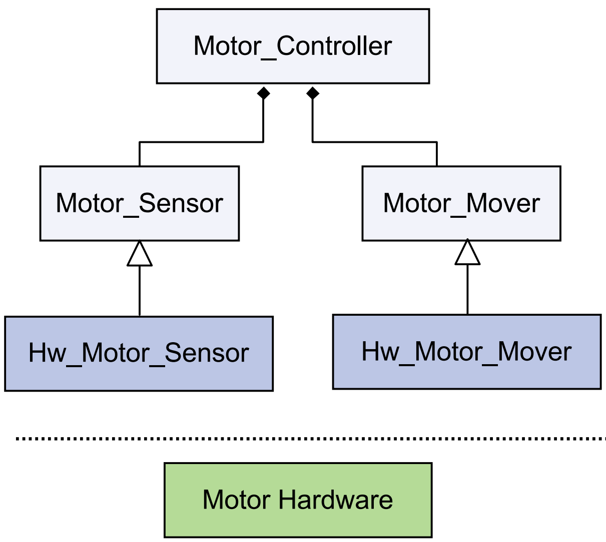


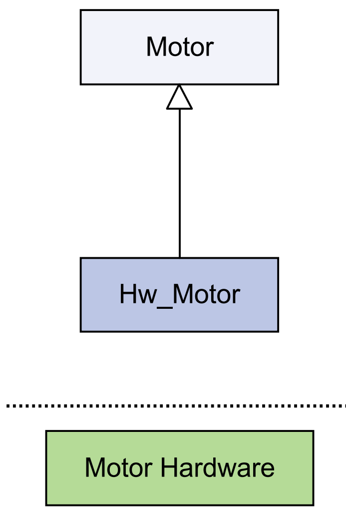
Guideline III: Interoperable Transformations



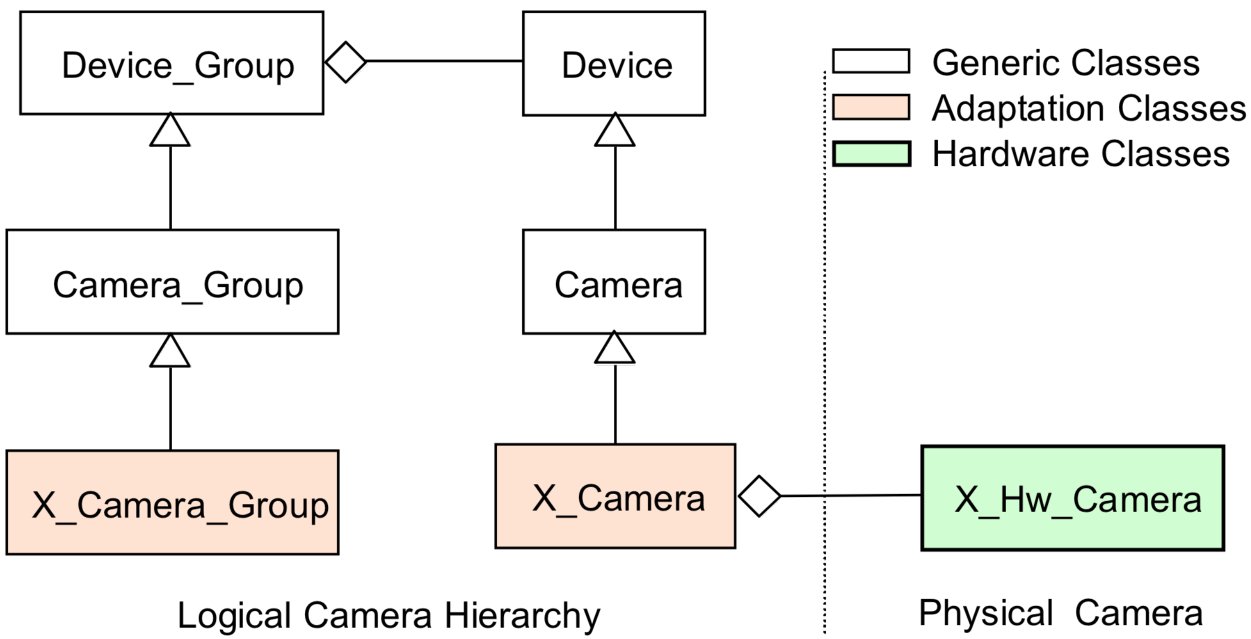
Guideline IV: Abstract Model



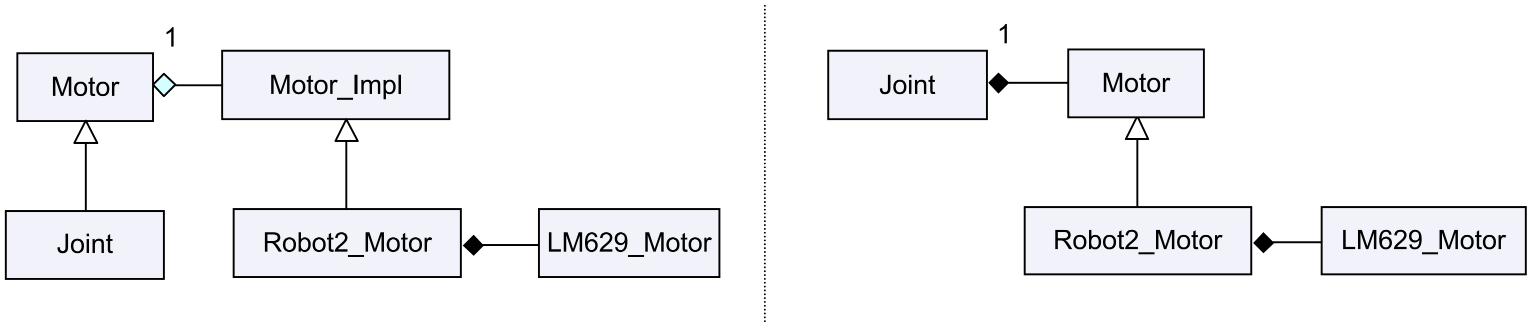




Guideline V: Separating Logical from Physical Hierarchies



Guideline V: Separating Logical from Physical Hierarchies

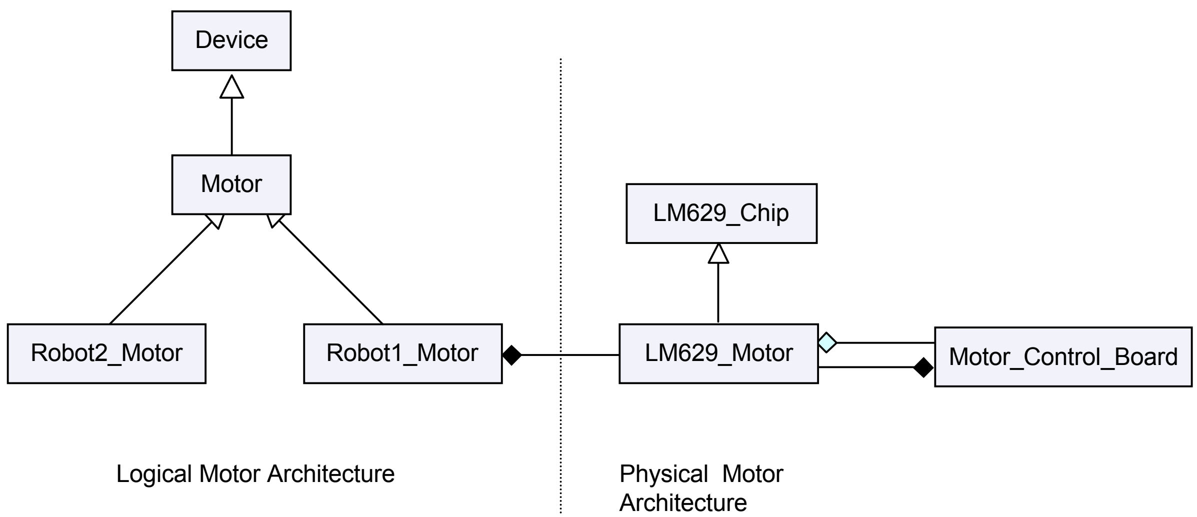


(a) Joint is a Motor

(b) Joint has a Motor

Revision 2

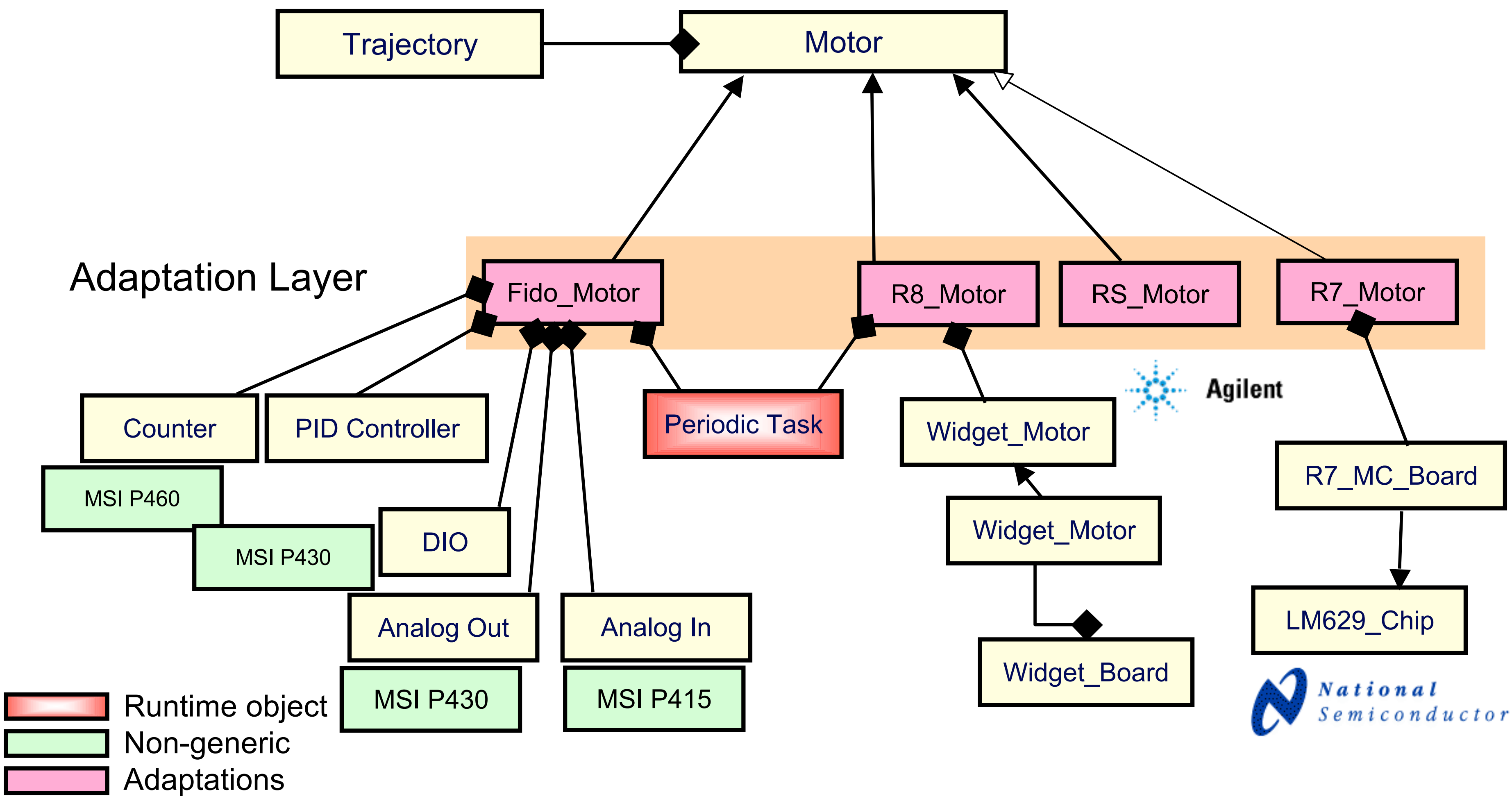
Revision 3



Logical Motor Architecture

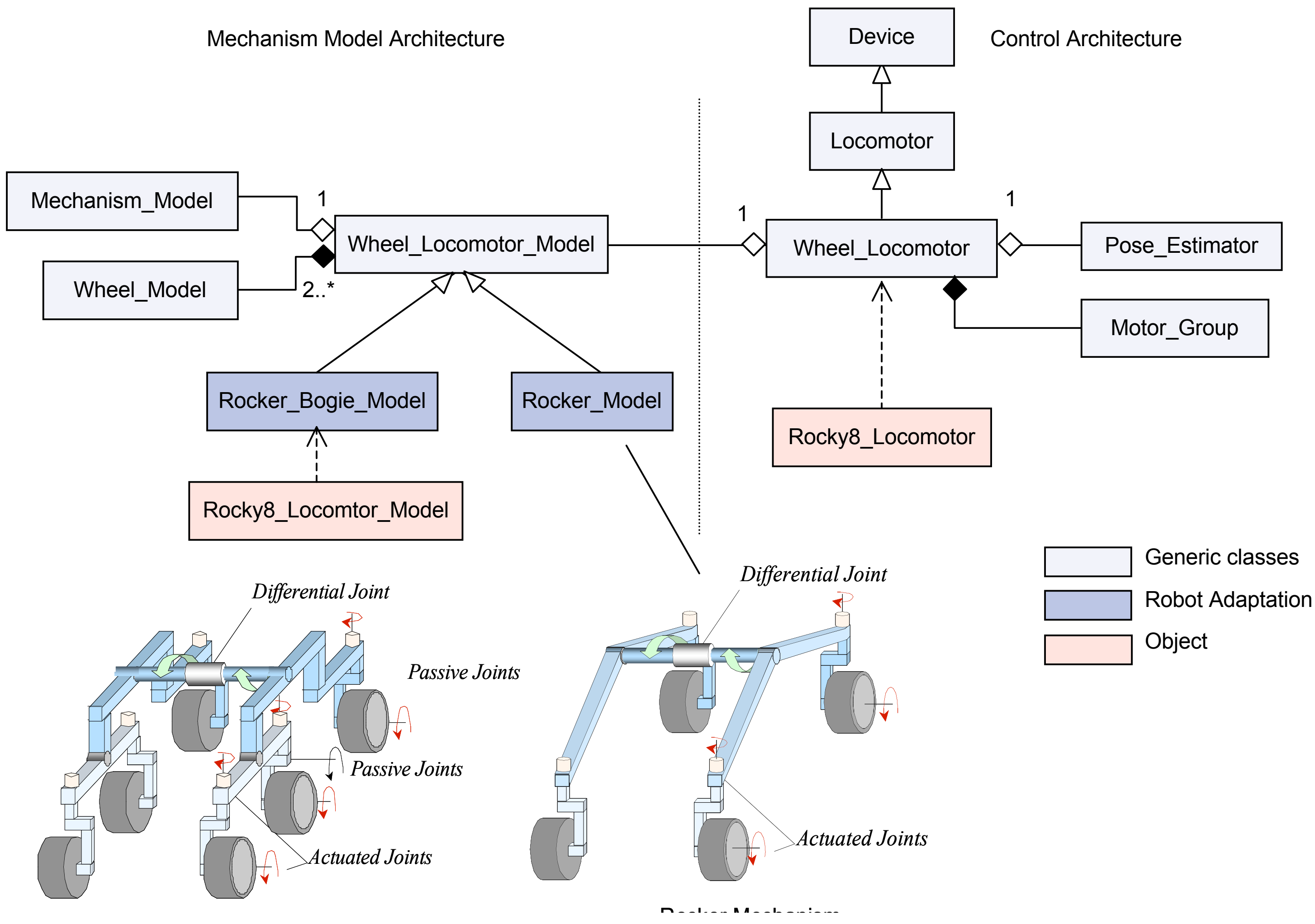
Physical Motor Architecture

Guideline IX: Encapsulate Run-time Model

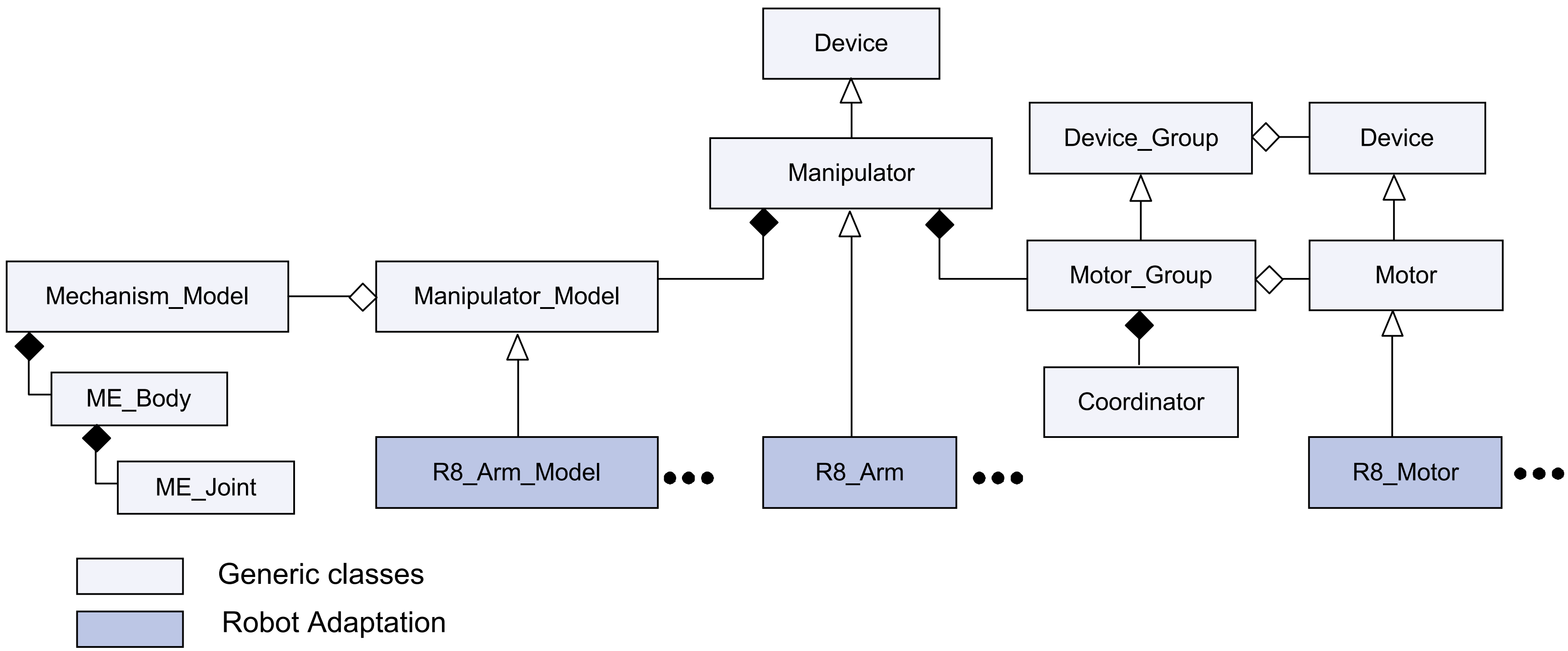


Guideline VI: Separating Models from Control

Separating model and control



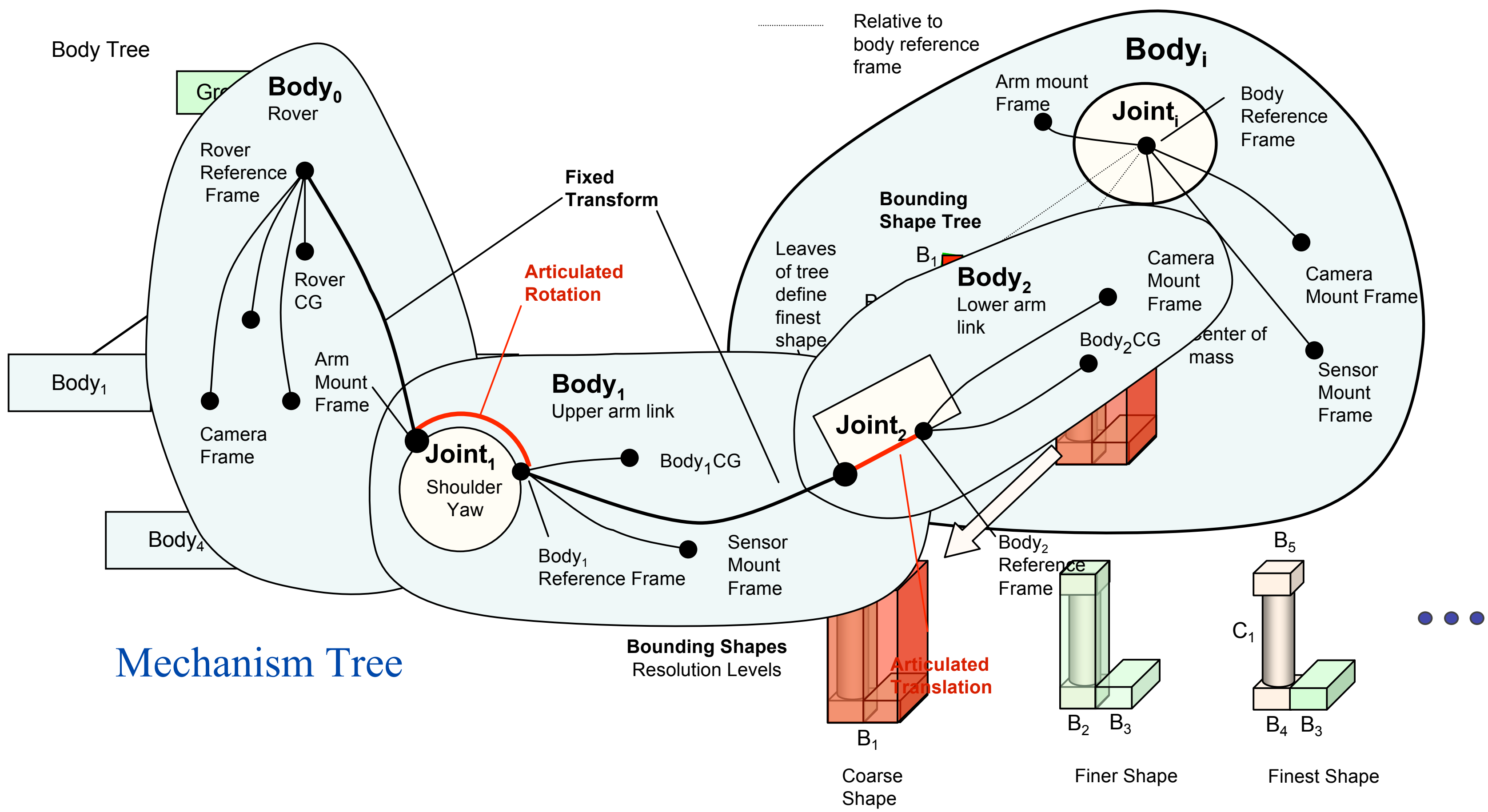
Guideline VI: Separating Models from Control



Guideline VII: Unifying Mechanism Model

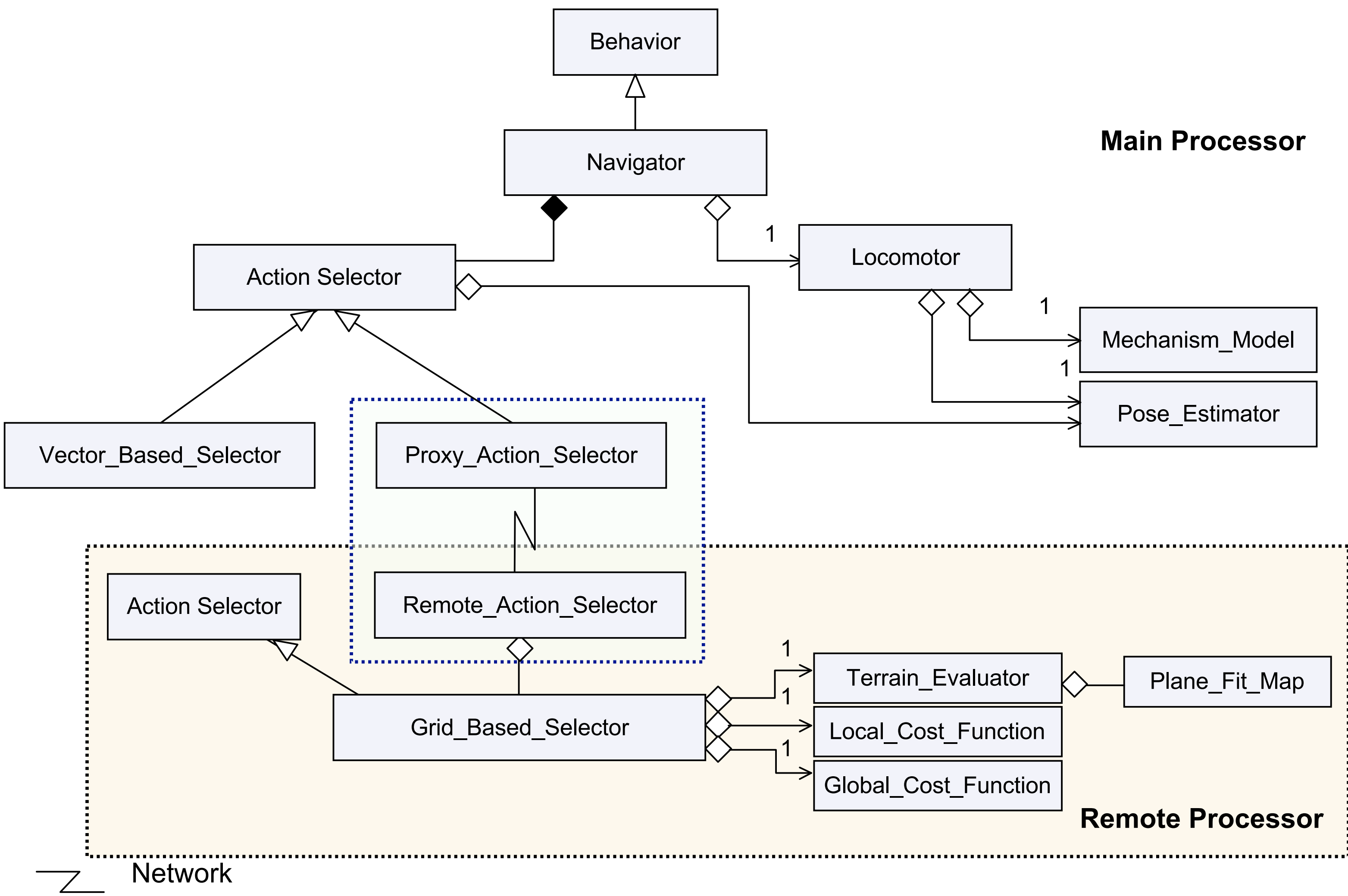
Unifying mechanism model

Bodies and Joints

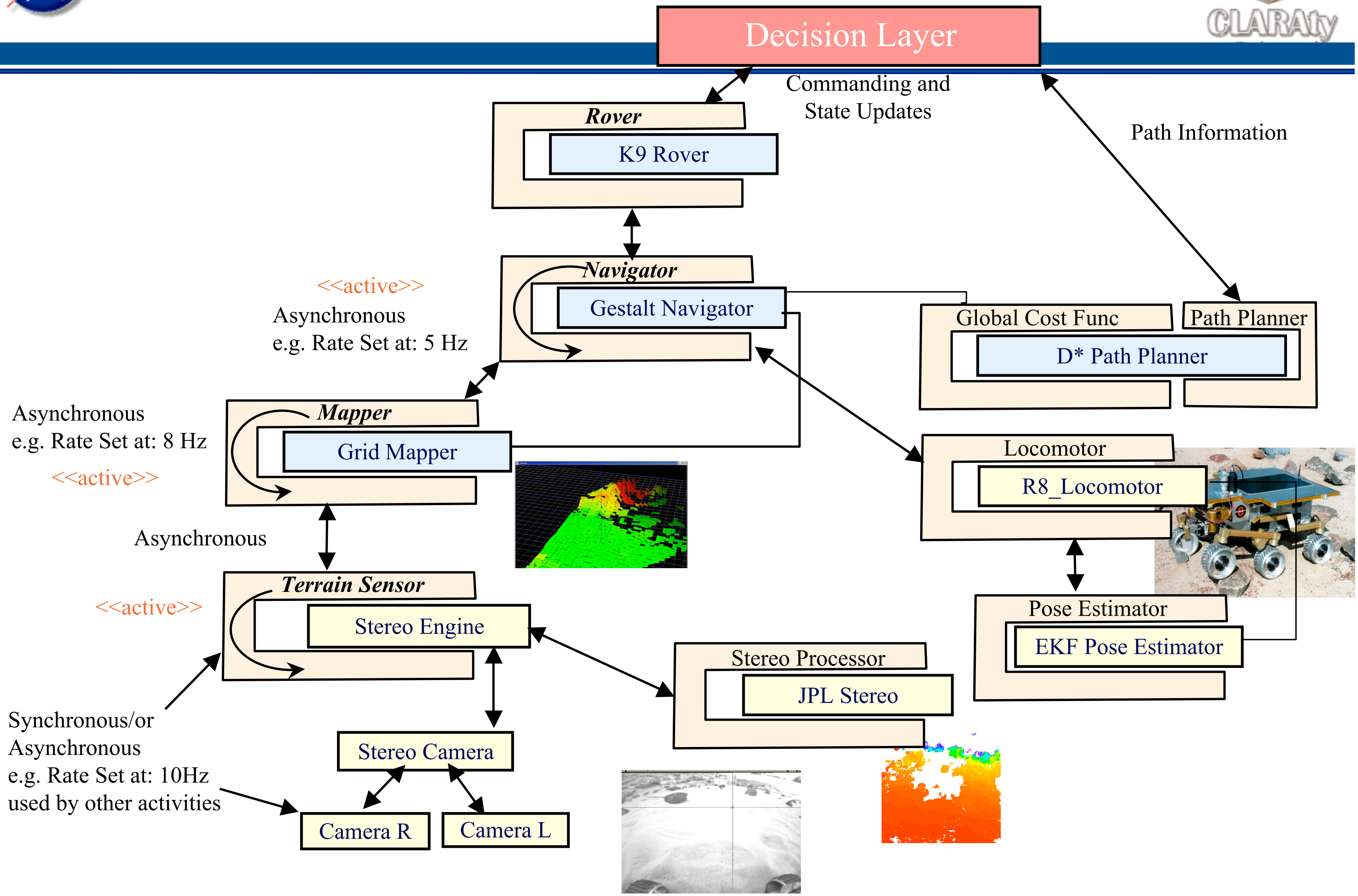


Mechanism Tree

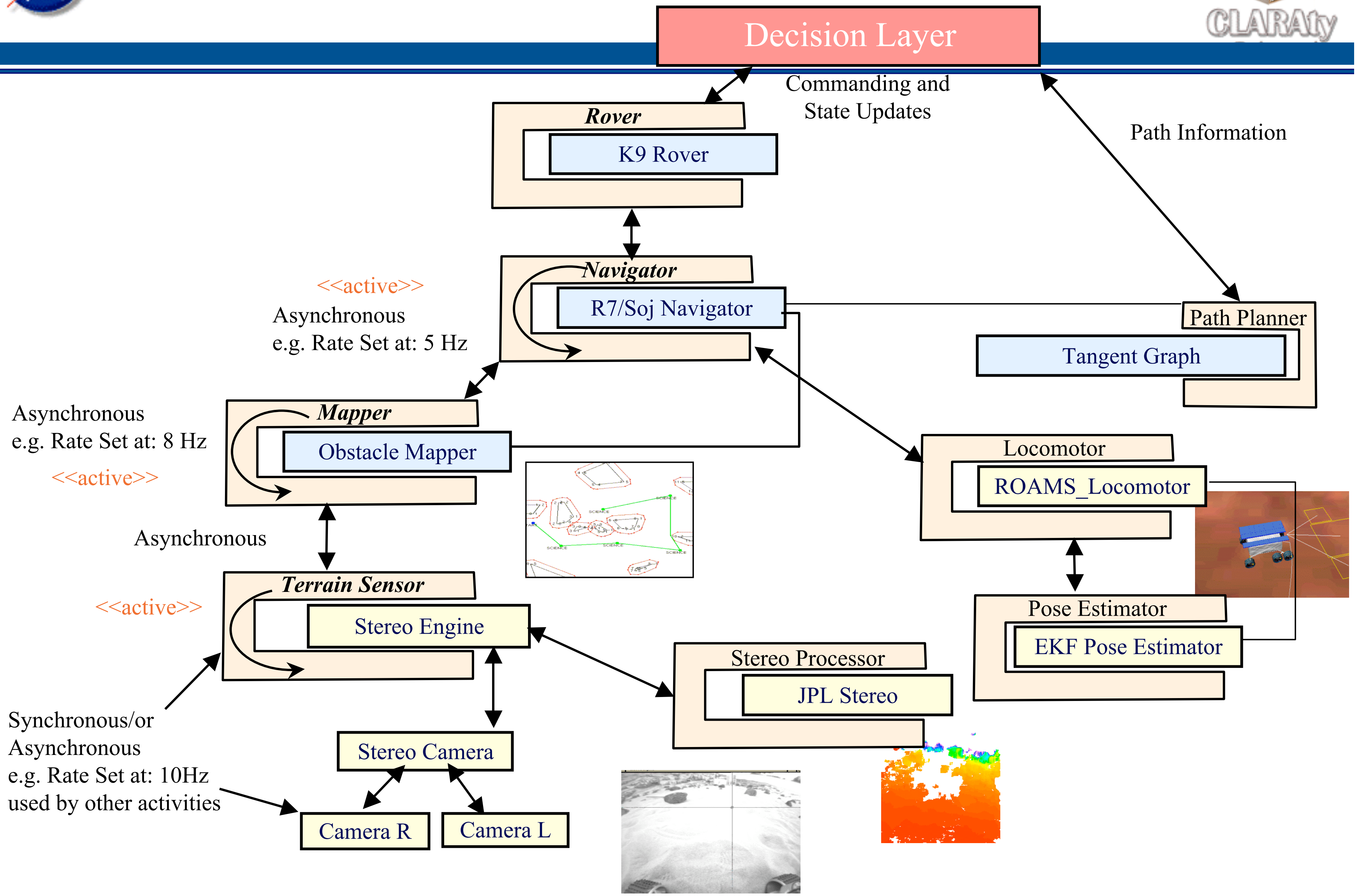
Guideline VIII: Separating Interface from Implementation



Putting it All Together - Swapping Navigation Algorithms



Putting it All Together - Swapping Navigation Algorithms



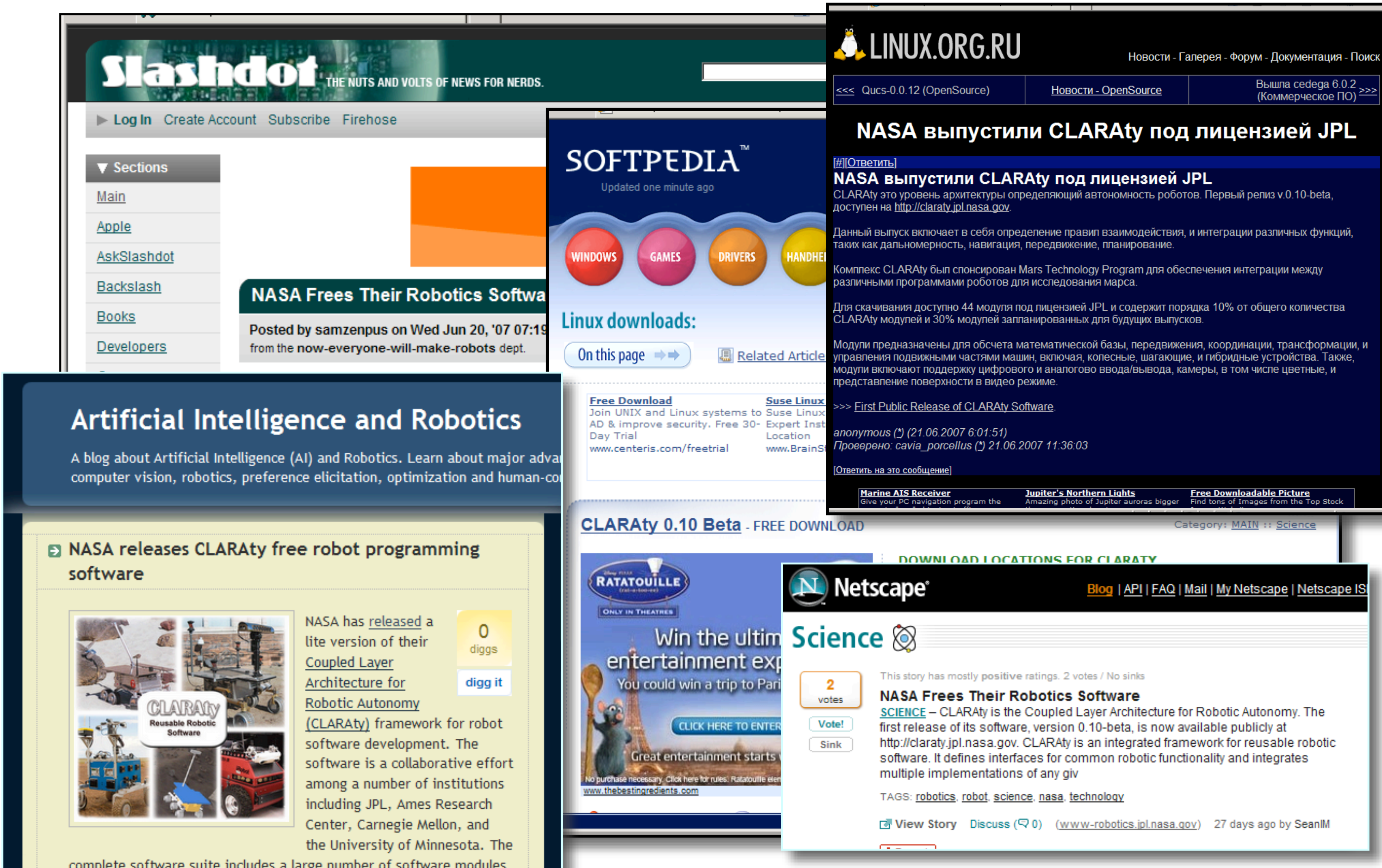
Status and Future Plans

Product: CLARAty Modules

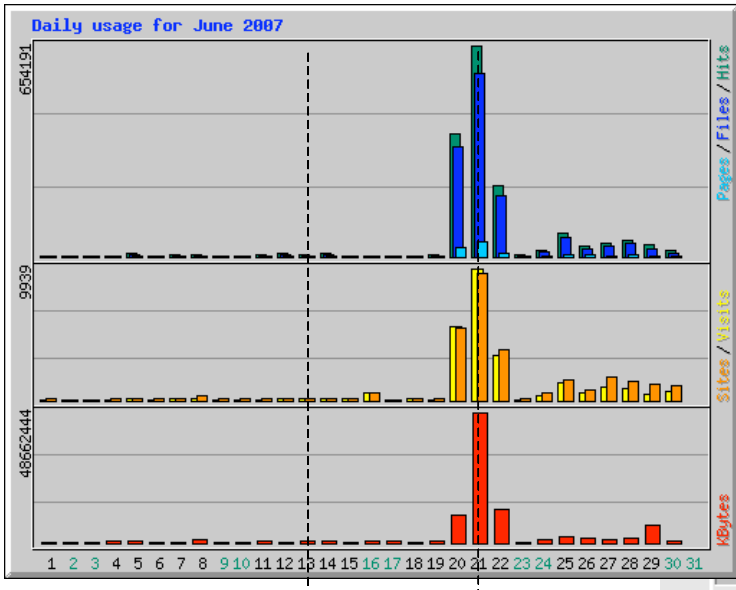
- 425 modules in repository
- 54 modules are technology contributions (~13%)
- 870K lines of logical C++/C (300K from MER)
  - Major increase due to incorporation of MER FSW and navigation
- Six rover adaptations
  - K10, Rocky 8, FIDO, K9, Rocky 7, ATRV, Athena, and DistAv Pluto
- 44 modules released in the public repository
- 4 are ITAR restricted
- 25 may be IP restricted
- remaining modules are ITAR clear and planned for public release

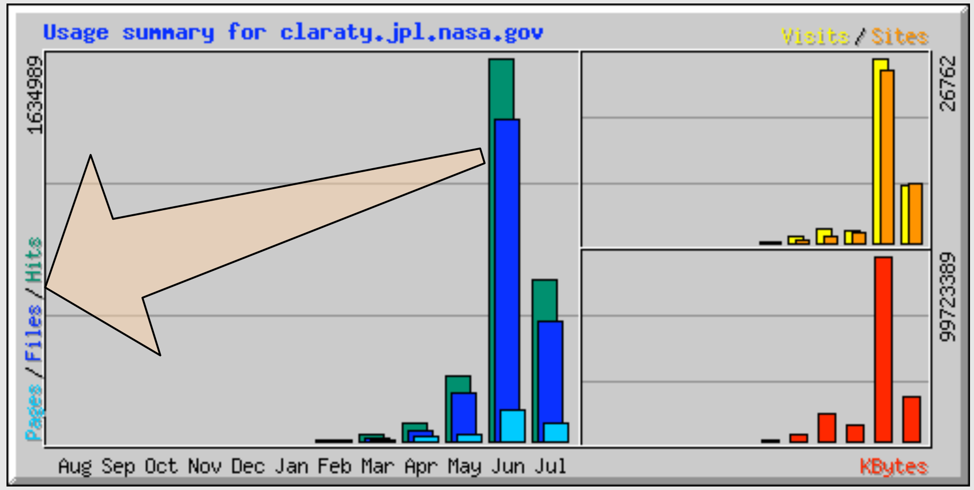
CLARAty Integration Levels
•Level I – Deposited
•Level II – Encapsulated
•Level III – Refactored
•Level IV – Formally reviewed
•Level V – Publicly released

Sample Coverage of CLARAty Release



CLARAty Release Statistics





CLARAty Released

Slashdot Article

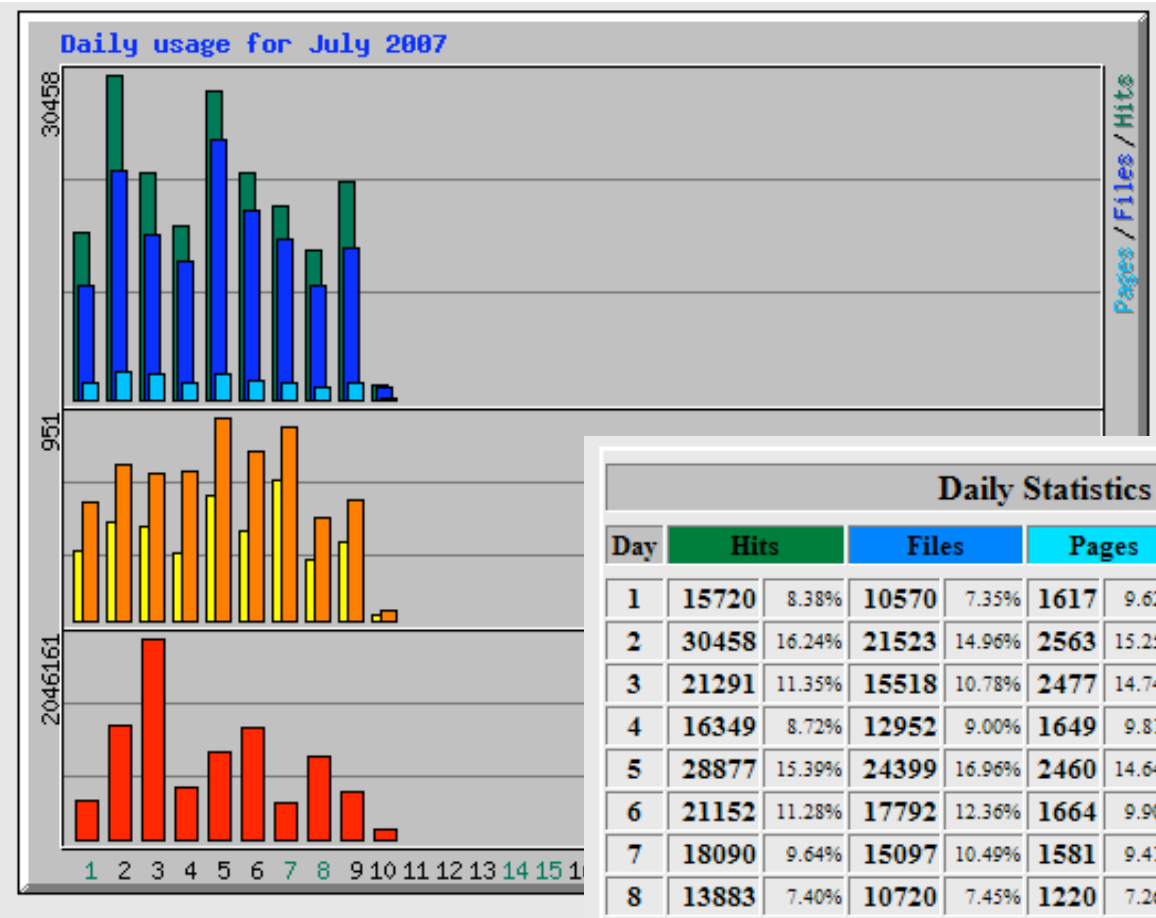
Summary by Month

Month	Daily Avg				Monthly Totals					
	Hits	Files	Pages	Visits	Sites	KBytes	Visits	Pages	Files	Hits
Jul 2007	18757	14389	1680	388	4765	7940189	3885	16803	143891	187574
Jun 2007	54499	45656	4451	892	25123	99723389	26762	133547	1369692	1634989
			793	58	1403	8890087	1821	24586	202490	278946
			555	67	936	14531668	2022	16650	47578	75654
			297	59	457	3127604	837	4169	15540	27010
			132	36	113	238984	108	398	2818	5829
						134451921	**35435**	**196153**	**1782010**	**2210002**

Stats

- Strong interest among general public and robotics community.
- Tens of web sites carried articles on the release
- Articles written in multiple languages and hosted world wide
- CLARAty site received around **2.5 million** hits in two months
- Total of **3,300 downloads** of the software in one month
- Hundreds of comments from the public

CLARAty Web Site Statistics:
Stable Increase in Traffic – July 2007



Daily Statistics for July 2007												
Day	Hits		Files		Pages		Visits		Sites		KBytes	
1	15720	8.38%	10570	7.35%	1617	9.62%	327	8.42%	549	11.52%	403881	5.09%
2	30458	16.24%	21523	14.96%	2563	15.25%	464	11.94%	730	15.32%	1147833	14.46%
3	21291	11.35%	15518	10.78%	2477	14.74%	441	11.35%	686	14.40%	2046161	25.77%
4	16349	8.72%	12952	9.00%	1649	9.81%	317	8.16%	698	14.65%	538035	6.78%
5	28877	15.39%	24399	16.96%	2460	14.64%	583	15.01%	951	19.96%	895687	11.28%
6	21152	11.28%	17792	12.36%	1664	9.90%	416	10.71%	789	16.56%	1131185	14.25%
7	18090	9.64%	15097	10.49%	1581	9.41%	661	17.01%	909	19.08%	362198	4.56%
8	13883	7.40%	10720	7.45%	1220	7.26%	284	7.31%	476	9.99%	837820	10.55%
9	20370	10.86%	14283	9.93%	1479	8.80%	372	9.58%	567	11.90%	487337	6.14%
10	1384	0.74%	1037	0.72%	93	0.55%	27	0.69%	50	1.05%	90052	1.13%

Summary

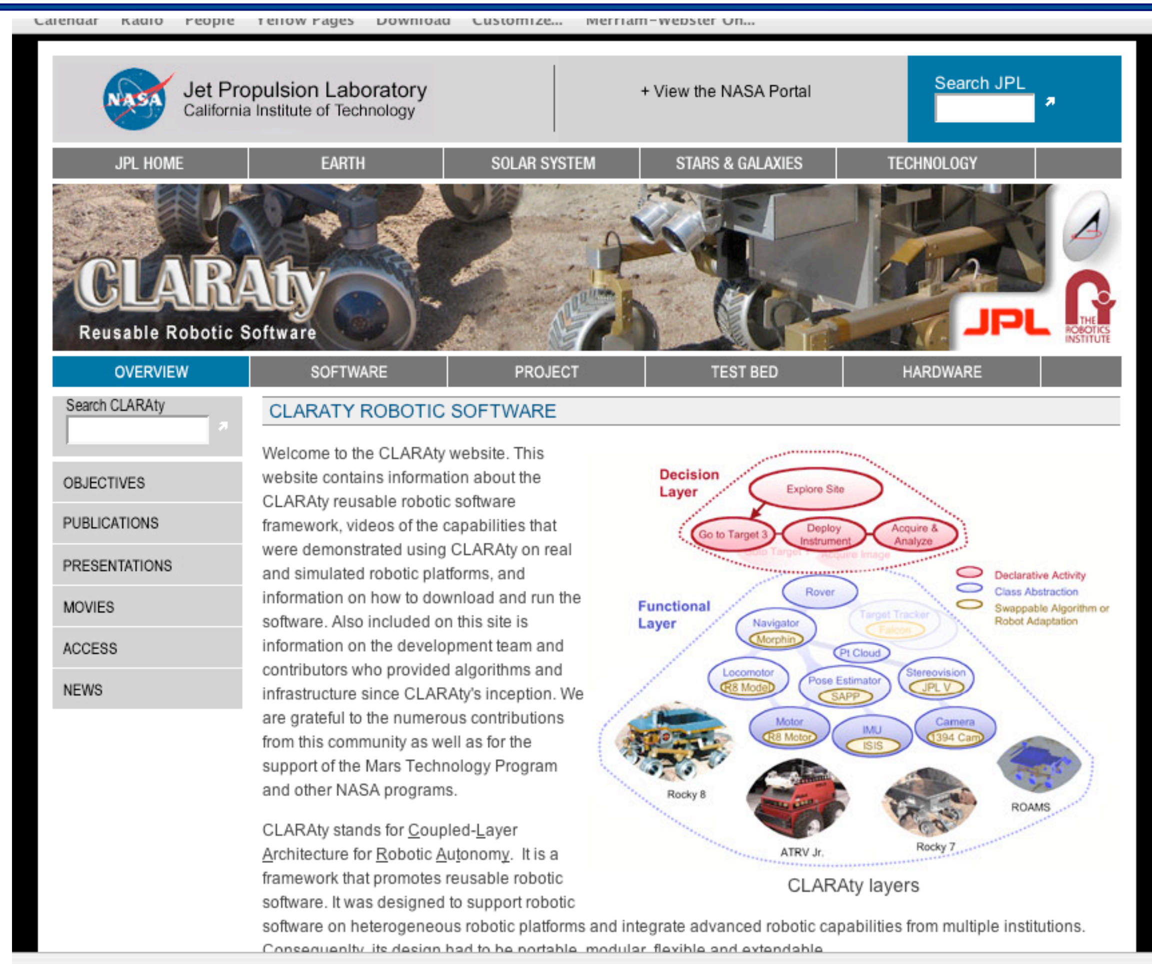
- Presented a quick overview of the CLARAty effort toward increasing the reuse of robotic software
- Deployed at multiple institutions
- Deployed on multiple heterogeneous robots
- Integrated multiple technologies from different institutions
- Delivered algorithms for formal validation
- Enabled new technology developments on multiple platforms
- Integrated flight algorithms for detailed performance characterization and operation on research rovers.
- Taking a technology from inception, to development in CLARAty, to validation, and now to integration into flight
- Completed the first public release and interest from the community looks promising

Acknowledgement: CLARAty Developers
(Core Team)

- NASA Ames Research Center
  - Lorenzo Flueckiger *(Co-I)*
  - Hans Utz
- Carnegie Mellon
  - Reid Simmons *(Co-I)*
  - Nick Melchior
  - David Apfelbaum
- University of Minnesota
  - Stergios Roumeliotis *(Co-I)*
  - Anastasios Mourikis
  - Nikolas Trawny
- Jet Propulsion Laboratory
  - Issa A.D. Nesnas *(PI)*
  - Tara Estlin *(PI - DL)*
  - Hari Das Nayar
  - Mihail Pivtoraiko
  - Richard Petras
  - Michael McHenry
  - Daniel Clouse
  - Dan Gaines
  - Robert Steele
  - Khaled Ali
  - Won Kim
  - Kelly Breed
  - Jeffrey Edlund
- Key Former Developers
  - Richard Volpe
  - Anne Wright
  - Max Bajracharya
  - Clay Kunz
  - Antonio Diaz Calderon
  - Chris Urmson
  - Richard Madison
  - I-Hsiang Shu
  - Randy Sargent

For the complete list of key former developers and contributors see:
http://claraty.jpl.nasa.gov -> Project -> Team

Team Received One NASA Center Best Award in 2006



Thank you